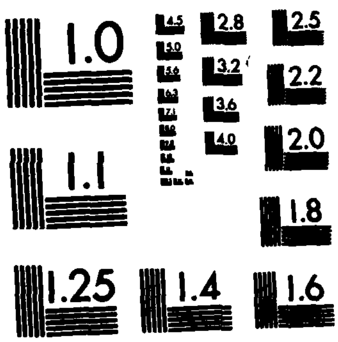
AD-A124 598 SPECIFICATIONS OF A SIMULATION MODEL FOR A LOCAL AREA NETWORK DESIGN IN S..(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA I T MASTROCOSTOPOULOS OCT 82 1/2

UNCLASSIFIED F/G 9/2 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

2

NAVAL POSTGRADUATE SCHOOL

Monterey, California

DEPARTMENT OF THE NAVY NAVAL POSTGRADUATE SCHOOL

AD A 124598

THESIS

SPECIFICATIONS OF A SIMULATION MODEL
FOR A LOCAL AREA NETWORK DESIGN
IN SUPPORT OF STOCK POINT LOGISTICS
INTEGRATED COMMUNICATIONS ENVIRONMENT (SPLICE)

by

Ioannis Th. Mastrocostopoulos

October 1982

Thesis Advisor: N.F. Schneidewind

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. A124 598	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Specifications of a Simulation Model for a Local Area Network Design in Support of Stock Point Logistics Integrated Communications Environment (SPLICE)		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis; October 1982
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Ioannis Th. Mastrocostopoulos		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE October 1982
		13. NUMBER OF PAGES 100
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

SPLICE
Local Area Network (LAN)
Stock Point Logistics Integrated Communications Environment
Simulation model specifications

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This thesis gives the specifications of a simulation model for a particular Local Area Network (LAN) system which implements functions of the Stock Point Logistics Integrated Communications Environment (SPLICE). First, system simulation and LAN components and performance measures are discussed in general. Then, the components of a LAN system model

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

#20 - ABSTRACT - (CONTINUED)

employing bus architecture are identified and modeled as an open network of queues.

A

Approved for public release; distribution unlimited.

Specifications of a Simulation Model for a Local Area Network Design in Support of Stock Point Logistics Integrated Communication Environment (SPLICE)

by

Ioannis Th. Mastrocostopoulos
Major, Hellenic Army
B.S.E.E., Naval Postgraduate School, 1982

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
October 1982

Author: ______________________

Approved by: ______________________ Thesis Advisor

Norman R Lyons
______________________ Second Reader

Uno R. Kodres
______________________ Chairman, Department of Computer Science

W. M. Woods
______________________ Dean of Information and Policy Sciences

ABSTRACT

This thesis gives the specifications of a simulation model for a particular Local Area Network (LAN) system which implements functions of the Stock Point Logistics Integrated Communications Environment (SPLICE). First, system simulation and LAN components and performance measures are discussed in general. Then, the components of a LAN system model employing bus architecture are identified and modeled as an open network of queues.

TABLE OF CONTENTS

LIST OF TABLES

TABLE #

LIST OF FIGURES

I. INTRODUCTION

A. GENERAL

Computer networks are an important part of our society and directly or indirectly affect many people. In recent years, there has been considerable interest in studies related to a special kind of network, the Local Area Networks (LAN), and it is reasonable to expect a proliferation of LANs within the next decade, due to new technologies and potential applications.

One major attraction of LANs is that they make possible the economical and efficient sharing of resources, i.e., processors, expensive peripheral devices, databases, communications bandwidth, information, etc. Because of the growing complexity of the Navy's logistics functions, the increase in the number of above mentioned resources, and the need to support interactive operations at the Navy's stock points and inventory control points, have led the developers of the Supply Point Logistics Integrated Communications Environment (SPLICE), to employ a local area network at each site for integration of all computer resources [Ref. 1]. Also this local area network will serve as the basic linkage for future system integration within a designated location [Ref. 2]. That is, as SPLICE requirements evolve and as technology changes, dissimilar devices e.g., new host computers, mass storage devices, teleprocessing gateways, can be added to the LAN easily without having to redesign other parts of the SPLICE system.

The SPLICE project at the Naval Postgraduate School will produce specifications and recommendations for the design of the LAN to be implemented at stock points and inventory control points. During the design phase of any system a simulation model has proven to be an effective analytical tool to assist in the decision making process. Simulation modeling, though based heavily upon computer science, mathematics, probability, and statistics, remains an intuitive process [Ref. 3]. The following quotation of R. E. Shannon [Ref. 3] stresses a note of warning about simulation: "Like all powerful tools which depend heavily upon art in their application, simulation is capable of giving either very good or very bad results, depending upon how it is used. It can either enlighten or mislead." This thesis covers work done toward the development of such an effective tool, a simulation model for local area networks design.

1. Scope of Research

Although simulation models differ significantly in their construction and use, the analysis and development of all types consists of three general phases: Conceptualization (specifications), implementation, and experimentation. Towards the development of a complete simulation model for a local area network implementing the SPLICE functions, this research covers the conceptualization phase by discussing the specifications of such a model.

2. Approach

The development of specifications of the LAN simulation model went through clearly definable but overlapping and frequently iterative steps. A brief discussion of each step taken in the development process follows:

a. Study system simulation in general: A general study of system simulation was conducted for the purpose of obtaining a knowledge of general system simulation and selecting an appropriate simulation technique to be used with the LAN simulation model.

b. Study LAN components and performance measures in general: LAN components and performance measurements were investigated in order to develop a knowledge and understanding of what actually composes a LAN and determine the different types of performance measures which could be made on computer networks in general. Such investigation was necessary for the development of an accurate LAN simulation model.

c. Giving specifications for a particular LAN simulation model: This third step in the development process was concerned with the detailed design of a LAN simulation model. That is, the symbolic (logical-mathematical) description of a local area network system to the level of detail appropriate to give a valid representation of the system.

d. Study model implementation and experimentation: This final step in the development process, though not within the scope of the thesis, was conducted in order to obtain a knowledge of these two important phases in simulation model construction. This knowledge was necessary in determining the level of LAN system resolution to be represented in the simulation model.

B. OVERVIEW

Following the steps taken in the development process of a simulation model, this thesis discusses, first, in Chapter II the development of a system simulation. Next, in Chapter III, are discussions of the various components which make up a local area network and different performance measures are described. Then, in Chapter IV, the specifications of a simulation model for a local area network are given. Finally, Chapter V is concerned with the implementation and experimentation phases of simulation model construction.

II. SYSTEM SIMULATION

One of the most important and useful tools for analyzing the design and operation of complex processes or systems is simulation. Simulation is a very wide-open and somewhat not well defined subject of great importance to those responsible for the design of systems as well as those responsible for their operation.

In general usage, simulation is defined as an act or process that gives the appearance or effect of some part of reality--a counterfeit, a feigning [Ref. 4]. Among the many definitions offered by various authors, the more suitable for the purposes of this thesis is given by Shannon [Ref. 3]:

> Simulation is the process of designing a model of a real system and conducting experiments with this model for the purpose either of understanding the behavior of the system or of evaluating various strategies (within the limits imposed by a criterion or set of criteria) for the operation of the system.

Thus, it is understood that the process of simulation includes both the construction of the model and the analytical use of the model for studying the system.

Topics relevant to a fundamental study of simulation are those that deal with and define the key words in the above definition: system, model. This chapter describes the basic concepts of what constitutes a system, how a model can be used to represent a system, and how a system and model description affect the development of system simulation.

A. SYSTEMS

Webster's New Collegiate Dictionary (1980, p. 1175) defines a system as "a regularly interacting or interdependent group of items forming a unified whole." For the purposes of this thesis a more specific and operational definition will be used, given by Fitzgerald [Ref. 5]:

> A system can be defined as a network of interrelated procedures that are joined together to perform an activity or to accomplish a specific objective. It is, in effect, all the ingredients which make up the whole. And a procedure is a precise series of step-by-step instructions that explain
>
> - What is to be done.
> - Who will do it.
> - When it will be done.
> - How it will be done.

Systems are distinguished from one another by their static and dynamic structures. The entities (i.e., a job) that make up a system, along with their associated attributes (i.e., job priority) and membership relationships (conncection between entities) define its static structure. The activities in which these entities engage specify its dynamic structure. The relationships between the attributes of a given entity are called functions and very often are being given in some form of mathematical expression. Collectively, the attributes (i.e., their values) of an entity define its states, and the states of the entities of a system define the state of the system. These entity states can be viewed from both a static or dynamic system standpoint and provide a significant tool for examining system behavior. That is, they can be viewed at a single point in time or over a series of points in time.

Every system has three basic features. It has an environment in which it exists. It has a set of boundaries which distinguish the system from the rest of its environment. And it has a set of subsystems which are its components parts [Ref. 6].

1. Objectives for Analyzing a System

The objective in studying system behavior are to learn how the state transitions occur, to predict transitions in state, and to control state transitions. One way in which these objectives can be satisfied is through the evaluation of alternatives [Ref. 6]. An evaluation of alternatives approach is concerned with the relationship between system inputs, which induce transitions in state and system outputs which measure these transitions in state.

There are three common properties to the evaluation of the alternative approach: The first is a straightforward analysis in which the system and its inputs are specified and the outputs are then measured. The second is broader in purpose and can be used to evaluate the relative merits of alternative system design when input is given and certain desirable characteristics for the output are specified. The third and final variant is when the system is specified and an effort is made to determine the input that produces the desired output. In this research, a combination of the second and third approaches is used.

2. System Classification

Systems can be classified in a number of ways [Refs. 4,5,6]. Unfortunately, none is completely satisfactory, although each serves a particular purpose. Some of these classification schemes are as follows:

a. Natural vs Man-made

The distinction between the two is obvious.

b. Open vs Closed

The distinction here is not so obvious. A closed system is one which automatically controls or modifies its own operation by responding to data generated by the system itself, thus it is one which can exist in a number of alternative environments. An open system, on the other hand, is one which does not provide for its own control or modification, thus it is one which only exists in a particular environment.

c. Discrete vs Continuous

The distinction between the two depends upon the way that they change from one state to another. The distinction can best be seen by considering the values that can be taken on by the variables that characterize the state of the system [Ref. 4]. Continuous systems include variables that can take on any real value in a prescribed interval or intervals. Discrete systems include variables that can take on only one particular value from among a finite (but possibly very large) set of values.

d. Deterministic vs Stochastic

The distinction between the two depends upon the causal relationship between input and output. The output of a deterministic system can be predicted completely if the input and the initial state of the system are known. That is, for a particular state of the system, a given input always leads to the same output. A stochastic system in a given state, on the other hand, may respons to a given input with any one among a range of distribution of outputs. For a stochastic system--given the input and the state of the system--it is possible to predict only the range within which the output will fall and the frequency with which various particular outputs will be obtained over many repetitions of the observation. It is impossible to predict the particular output of a single observation of the system.

e. Adaptive vs Non-adaptive

As their names imply, these systems either can or cannot react to changes in their environment.

The above distinction takes on real significance when the system is being analyzed. Conclusions reached about an open system must be carefully qualified in terms of the system environment. Also the analysis of an adaptive system requires a complete description of how the system environment causes changes in system state.

3. System State

The state of a system is determined by the values of the attributes of the system entities. Depending on the view concerning activities, whether they interact at discrete points or over periods of time, there can be attributes associated with dynamic as well as static system structures. That is, an activity can be fully or partially completed, be in progress or terminated, be waiting for another activity to occur or be interrupting another activity, etc. In a general view, there is no conflict in the description of state as a static or dynamic phenomenon, since at all times, whether between state changes in a static structure, or at system activity interactions in a dynamic structure, the concept of a system state is completely defined.

Viewed at a point in time, a system is always in one of a number (perhaps ennormous) of states. Viewed over a period of time, a system passes through a succession of states as its entities undergo system activities, change their attribute values and relationships, and become eligible for subsequent activities. Thus, the analysis of a given system is the study of its transitions in state as time passes.

The state transition of a system is described by two attributes (i.e., their values): magnitude and delay [Ref. 6]. The magnitude of a state transition refers to the absolute difference in an attribute's value over a specified period of time compared to its value before the state changes. The delay associated with a transition of state refers to the passed time between the arrival of an input and the actual transition of state caused by the input. Delay determines that an input can cause a transition state immediately upon its arrival, at some time after its arrival, or over a period of time following its arrival (continuously).

4. System Response

When viewed together, the magnitude and delay in a transition of state describes system response. That is, the way in which the system reacts to a given input. Five possible system responses are shown in Figure 2.1 [Ref. 6]. A stable system response is shown in Figures 2.1a, 2.1b, and 2.1c. In a stable response, the system moves over some finite period of time to a permanent new state of equilibrium (steady state) of finite value as a result of a single input to the system.

Two definitions are important in considering when to begin to measure output in a simulation experiment:

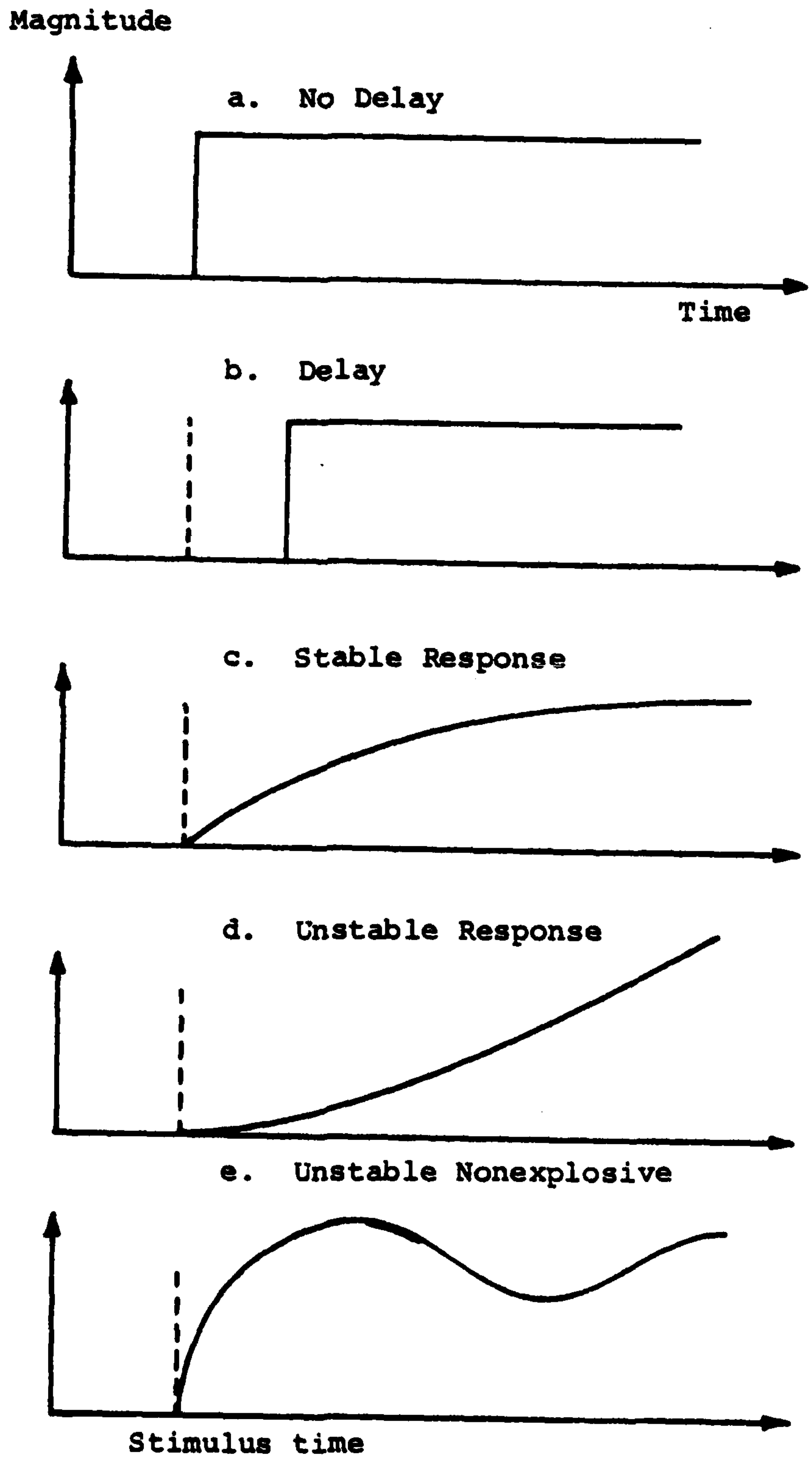


Figure 2.1. Possible System Responses

a. The system is said to be in the empty or idle state, if the initial value of the state variable under consideration is zero.

b. The system is said to be in a transient state, if it is in the process of moving from one steady state to another.

Results, as a general rule, should be taken only after a steady state is reached, following the transition period from an initial empty or idle system state. This reduces the influence of the systems behavior during the transition period on the overall evaluation of the system. An unstable system response is shown in Figures 2.1d, and 2.1e. Figure 2.1d exhibits an explosively unstable response where the value of the state variable continues to increase with the time without convergence to a new finite value. Figure 2.1e shows a non-explosive unstable system in which an equilibrium level exists but where system response oscillates around this level but does not converge to it.

5. System Performance

Earlier, three objectives were given for analyzing a system. They were to learn how system state transitions occur, to predict transitions in state, and to control transitions in state. These specific objectives for system analysis result from a desire or need to improve system performance. In turn, system performance is reflected in the sequence of states that a system undertakes over a given period of time. Normally, these sequences of system states are manipulated in some way by the system analyst to provide a series of performance measures.

The idea of what composes a performance measure varies with the system. For example, in Chapter III of this thesis the essential measures of Local Area Networks (LAN) performance will be discussed. These measures would have little or no significance if they were applied to the evaluation of another system (e.g., Water resources system).

In conducting the analysis of a complex system, it is often desired to obtain only a "feel" for system performance. It is exactly this type of analysis effort that can greatly benefit from a careful consideration of which performance measures will provide the needed "feel" or understanding of the system. Failure to devote adequate time to determine the performance measures that will be used can lead to confusion and delay in the analysis process.

6. System Optimization

One objective in analyzing a system and in measuring its performance is to optimize system performance, that is, to improve the effectiveness of the system. Generally, this system optimization involves the identification of certain critical system aspects and the development of certain procedures to control these aspects. Usually, however, a number of these critical system aspects are beyond the analyst's observation. Such aspects put constraints on system behavior and prevent total system optimization. In this case the final objective must be modified to one of optimizing performance under certain constraints. As it is often the case with the large complex systems, optimization of the entire system is virtually impossible, due to the size of the system and the complexity of the relationships among the various subsystems. Thus, optimization can and must occur with regard to individual subsystems or selected groups of subsystems.

B. MODELS

The initial step in analyzing a system beyond defining system components and identifying possible performance measures, is to build a model of the system.

A model is a representation of an object, system, or idea in some form other than that of the entity itself. Its purpose is usually to aid in explaining, understanding, or improving a system [Ref. 3].

1. Function of Models

The concept of representing some object, system, or idea with a model, is so general that it is difficult to classify all the functions that models perform. Fishman [Ref. 6], recognizes that a model performs the following functions:

a. Enables an investigator to organize his theoretical beliefs and empirical observations about a system and to deduce the logical implications of this organization.

b. Leads investigator to improved system understanding.

c. Brings into perspective the need for detail and relevance.

d. Expedites the speed with which an analysis can be accomplished.

e. Provides a framework for testing the desirability of system modifications.

f. Is easier to manipulate than the system is.

g. Permits control over more sources of variation than direct study of a system would allow.

h. Is generally less costly.

Summarizing the above functions, a model may serve one of two major purposes: descriptive, for explaining and/or understanding, and prescriptive, by predicting and/or duplicating behavior characteristics.

2. Model Classification

Models can be classified in a variety of ways and can take many forms. A model can be:

a. Iconic like a scale model airplane. This model resembles the system being studied.

b. Analog like an electric circuit that behaves like a mechanical system. In this model a property of the real system is represented by a substituted property which often behaves in a similar way.

c. Symbolic like a set of equations. This model uses a symbol rather than a physical device to represent an entity of the system.

Iconic models are good visual aids but are usually not good for predicting or explaining the behavior of the systems. Symbolic models are good for prediction and explanation but offer little as visual aids. Thus, different models exist for different purposes. This thesis concerns itself with symbolic models, so a further classification of those models follows.

3. Symbolic Models

A symbolic model represents a system using mathematical equations and algorithms. Symbolic or mathematical models can be distinguished according to their characteristics into four classification schemes as follow [Refs. 6, 7, 8, 9]:

a. Analytical vs Numerical

In an analytical model it is possible to deduce the behavior of the system, directly from the system's mathematical representation. Kirchoff's law is an example of an analytical model. A numerical model implies that an exact deduction of the system's behavior is not feasible but numerical methods can provide descriptions of the behavior for certain system aspects as are defined in the numerical model. Numerical integration is an example of a numerical model.

b. Continuous vs Discrete

Continuous-change models are used to represent systems that consist of a continuous flow of information or material (e.g., flow of gas in a pipeline). Continuous models are usually represented by differential equations which describe rate of change of the variables over time. Discrete-change models represent systems in which changes in the state of the system are discrete (e.g., messages arriving at a node of a network). Discrete models are usually represented by queueing theory and stochastic processes.

c. Static vs Dynamic

A static model either does not take into consideration the passage of time or describes the state of a system at a specified point in time. On the other hand, a dynamic model explicitly recognizes the passage of time. A dynamic model may specify also the relationships between the various system states at different points in time.

d. Deterministic vs Stochastic

In a deterministic system model, all the entities of the system have fixed mathematical or logical relationships to each other. Thus, the behavior of the system is completely determined by these relationships. In a stochastic model, part of the entity relationships, at least, vary in a random fashion. Thus, an analyst can at best, obtain an average description of a system behavior by using a stochastic model.

There are a number of different model descriptions possible, when the four sets of model characteristics are combined.

4. <u>Scope of a Model</u>

A model is used to predict results and describe the way the results are determined when a set of input conditions is given; that is, the analyst is concerned with both system response and system dynamics.

During the model building process the analyst must continuously deal with the problem of balancing the need for structural detail in describing the system, with the modeling resources and capabilities available. By its nature, a system model is a formalized abstraction of reality, thus the more structural detail it includes, the more it resembles the actual system. Additionally, increased modeling detail provides an increased capability for observing system response in a given system modification or series of system modifications [Ref. 10]. That is, increased detail provides more combinations of system modifications which can be made and more aspects of system response which can be measured.

While it seems to be desirable to include as much modeling detail as possible in a model, there are several problems which result from this increased detail. First, increased detail makes the modeling process more difficult. Second, it usually shifts the model characterization from analytical to numerical. Third, and most important, the analyst often does not understand the system to a degree that will allow specification of the system in the desired detail. Fourth, and final, the inclusion of increased detail may place an increased and unacceptable demand on analysis resources, i.e., time, personnel, facilities, etc.

Every type of system model must limit the amount of structural detail it includes. The degree to which this detail is limited must be determined through a process of balancing the original system analysis objectives against the analysis resources.

Although one of the purposes of building a system model is to improve the analyst's understanding of the system, there are three hazards associated with achieving this purpose [Ref. 6]: First, there is no guarantee that the results of the modeling effort will prove to be useful. Sometimes this type of failure is due to a lack of adequate resources, but more often, it is a case of an improper balance between available resources and the system analysis objectives. The second hazard deals with the analyst himself, who may think of his particular description of the system as the most accurate representation of reality, when, in fact, it is not. The third and most critical hazard involves the use of the model to analyze aspects of the system which the model was not intended to analyze.

C. DISCRETE EVENT SIMULATION

This section presents concepts applicable to the construction of a discrete event simulation model. An event denotes a transition in the state of a system entity. Thus, a discrete event simulation model can be defined as a model of system behavior in which entity state transitions are represented as a series of events occurring at discrete points in time.

Following is a discussion of event timing and entity-attribute relationships. The purpose of the discussion is to establish their importance in the realization of a discrete event system model. Additionally, alternative discrete event modeling techniques are discussed.

1. Event Timing

The actual approach to the discrete event modeling of a particular system depends on the nature of the event's inter-arrival rates. That is, whether these inter-arrival rates are deterministic or random in nature. If the inter-arrival rates are deterministic, then the modeling techniques used must reflect inter-arrival rates which vary according to a fixed relation or are equal. If the inter-arrival rates are random, then the modeling techniques must reflect their randomness.

With either type of inter-arrival rate, the occurrence of an event specifies a transition in the state of the system. The states of system entities remain constant between the occurrence of events, so there is no need to account for this dead time in a discrete event system model. When a particular event occurs and all the state transitions associated with the event are made, then simulated time is advanced to the time of the next event, where once again the required state transitions are made. This next event technique allows the analyst to compress time.

2. Entity-Attribute Relationships

There are two types of primary structural relationships which play a significant role in the modeling of a system. The first are the mathematical relationships which exist between the attributes associated with the various system entities. Sometimes the specification of the mathematical relationships for a system serve to describe completely the way in which system state transitions occur. The second are the logical relationships which exist between system components. A logical relationship checks to see if a certain condition holds. If it does, a given action is taken. If it does not hold, an alternative action is taken. Normally, a system is not modeled through the use of only one type of relationship, but through a mixture.

3. Alternative Modeling Techniques

There are three main ways of building discrete event system models [Ref. 6].

a. The event scheduling technique emphasizes the detailed description of the steps that occur during the processing of an event. Usually, an event naturally has a distinct series of steps associated with it.

b. The activity scanning technique emphasizes the review of all activities in a simulation to determine which can be initiated and terminated during the occurrence of a given event.

c. Finally, the process interaction technique emphasizes the continuous progress of an entity through the system. That is, from its arrival event to its departure event.

The development of the three techniques mentioned above has been associated directly with the development of discrete event simulation programming languages. In particular, GASP and SIMSCRIPT use the event scheduling technique, CSL uses the activity scanning technique, and GPSS and SIMULA use the process interaction technique [Ref. 4].

4. Queueing Models

The majority of discrete event simulation models are reducible to a series of queueing problems. In a queueing problem, an arrival event occurs and causes an entity to demand a service to be performed. The system responds to the demand for service either by performing it or by keeping the entity waiting (puts it in a queue) until it can perform the required service.

The objective in queueing oriented problems is usually to analyze how system performance varies in response to changes in system workload, system resource characteristics, or task selection schemes. In a given workload, system resources, and task selection must be resolved and explicitly specified in the simulation model.

III. LOCAL AREA NETWORK COMPONENTS AND PERFORMANCE MEASURES

A. GENERAL

A computer network, in general, is any system composed of one or more computers and associated terminals, communication devices, transmission facilities, and software/hardware to facilitate the flow of information between terminals and/or computers.

Metcalfe and Boggs [Ref. 11], distinguish three types of networks based on the parameters of bit rate and separation between computers:

Activity	Separation	Bit Rate
Remote networks	> 10 Km	< 0.1 Mbit/sec
Local networks	0.1--10 Km	0.1--10 Mbit/sec
Multiprocessors	< 0.1 Km	> 10 Mbit/sec

At one end of the above spectrum of activities there is a group of dissimilar computers tied together by a communication network, which enables a user to take advantage of a variety of computing resources, while at the other end there is an attempt to convert a collection of serial processors into parallel processors. Since local networks are near the middle of this spectrum, they may be built to gain the resource sharing of computer networking along with the parallelism of multiprocessing.

1. LAN Definition

As its name implies, a LAN links computing system components located within a geographically restricted area, such as a building or an office complex. There is no widely acceptable definition for a LAN. It should be noted that a panel discussion held during the Third Conference on Local Computer Networks--a conference devoted to develop a definition of a local computer network--failed to achieve a definition acceptable to all participants. A definition for local networks, which has been proposed by Franck [Ref. 12], is given below. He pictures a local computer network as consisting of three essential ingredients:

a. A high-speed transmission medium for data transmission over a "limited" distance. The nature of the transmission medium and the topology of the network are left unspecified.

b. Several network adapters attached to this transmission medium which serve as line interfaces for computing equipment. The adapters transmit data on the transmission medium.

c. Computing system components that can be attached to an adapter.

Franck's illustration of his definition is shown in Figure 3.1.

2. Overview

Based on the previous definition certain topics relevant to a fundamental study of the LAN components are those that deal with and define the key words: topology, physical transmission medium, and communication protocols for network management. To accomplish an effective network management is is essential that a series of basic transmission control

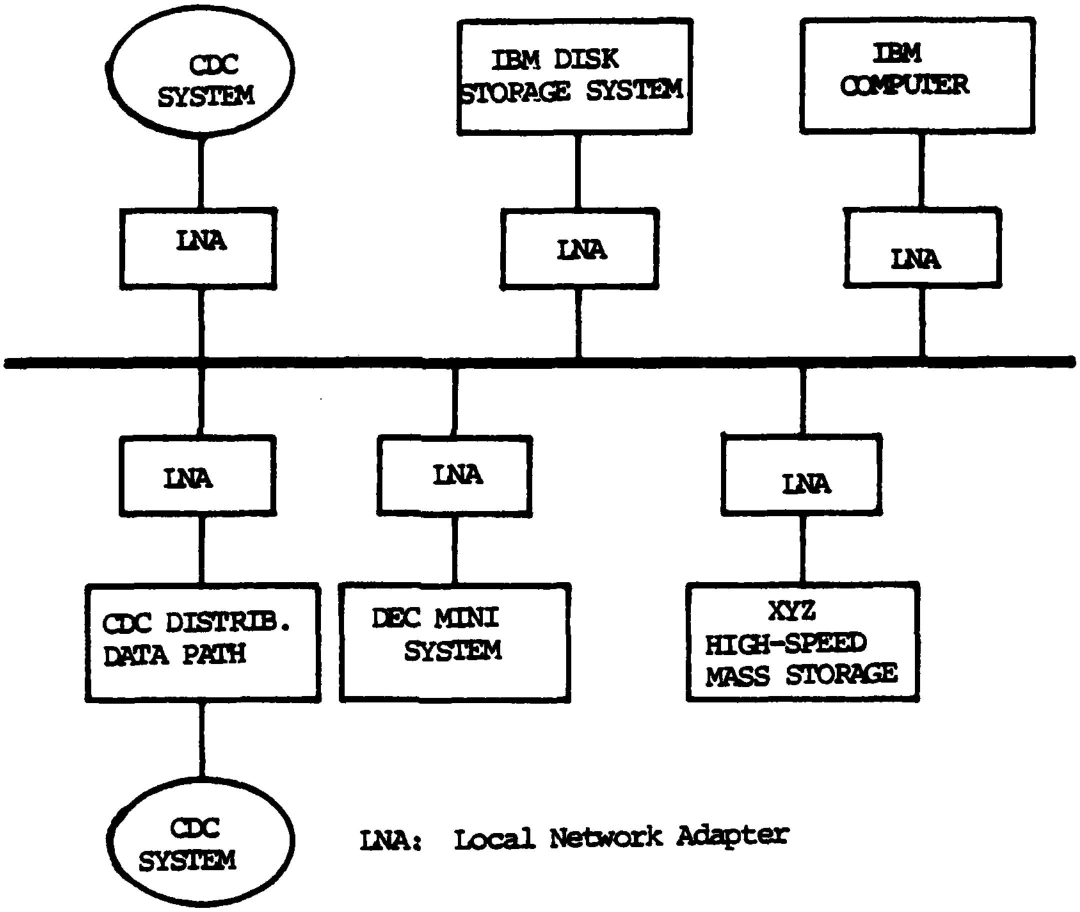


Figure 3.1. Local Computer Network

procedures be established to ensure the efficient and accurate transfer of information within the network. These procedures called network protocol, along with related aspects as channel access, flow control, and error control are extensively covered in Ref. (18) and not included here.

This chapter will discuss in general, the LAN topologies, the transmission medium, and the network performance measurements which are fundamental ingredients in any LAN analysis or design effort. It should be noted that the specific characteristics of the network components depend greatly upon the manner in which the LAN is implemented.

B. TOPOLOGY

Network topology is the spatial pattern formed by a network's digital devices, called nodes, and connecting links. Many characteristics of computer networks are determined or influenced by their topology. Some qualitative attributes can be inferred directly from the topology of the network independently of the particular implementation [Ref. 13]. Such attributes, among others, may be:

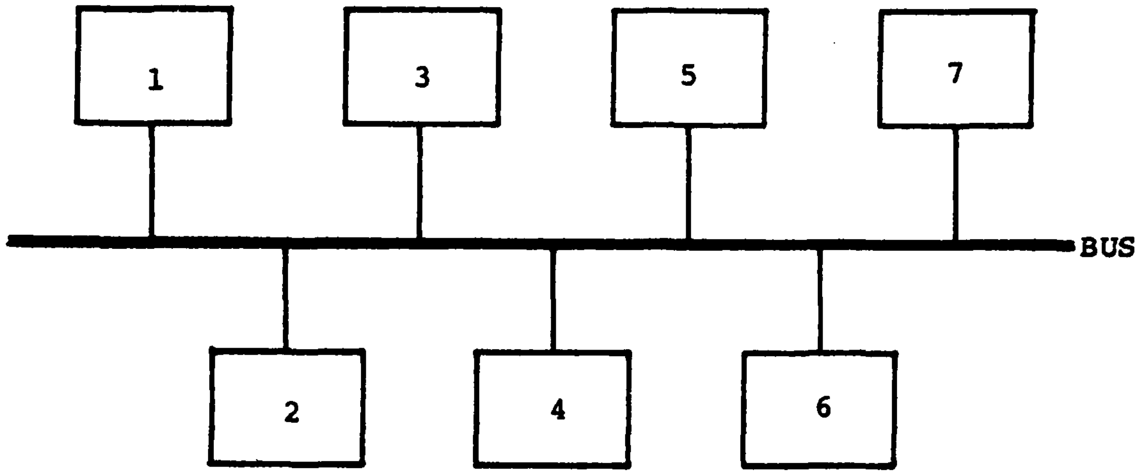
1. Modularity, the ability to make incremental changes in system capability.
2. Flexibility which measures the degree of freedom in adding a new element to the network.
3. The ability for graceful degradation.

As the number of network elements increases, the number of ways one can interconnect the various elements increases rapidly. Thus, it is not easy to classify all possible topologies for a network. This section will discuss only the basic topologies used in computer networks: Bus, ring, star, and mesh. The bus and the ring are characteristics of new decentralized network schemes intended for computer communications. The star is widely used in long-distance networks and in conventional (centralized) local networks, such as time-sharing systems. The mesh is characteristic of long-distance packet-switched (decentralized) networks.

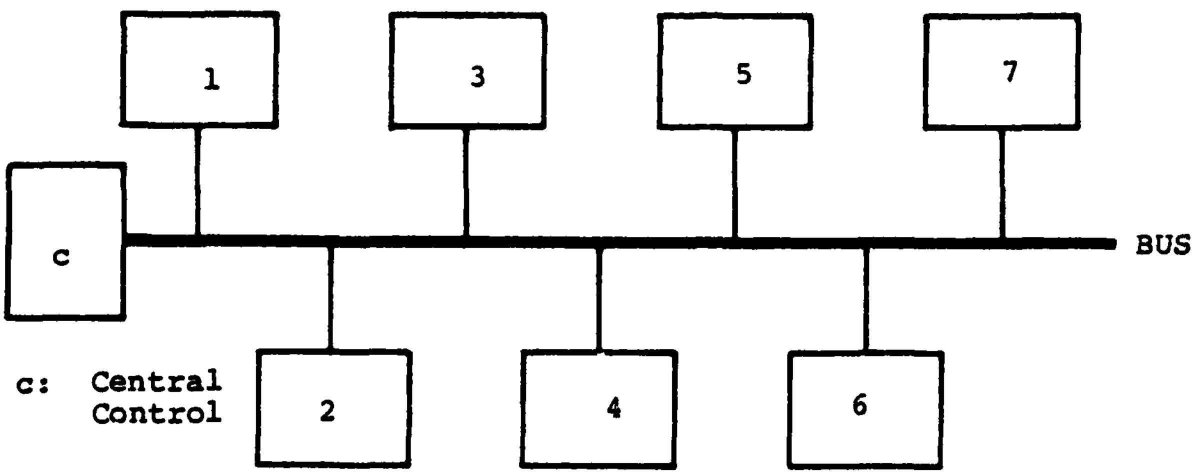
1. Bus Topology

The bus topology employs a shared transmission facility, called a bus, for information exchange between several nodes (computers, peripherals). Figures 3.2a and 3.2b illustrate distributed and centrally controlled bus topologies [Ref. 14]. In Figure 3.2a, all nodes are identical in nature but a technique must be developed to eliminate contention for the use of the bus. In Figure 3.2b, a single control node manages the traffic flows between every pair of nodes, i.e., all nodes must communicate with control node C before they set up a call.

The bus itself is employed in a broadcast mode, all nodes "hear" a message. The major advantage of a bus topology is its simplicity. It is easy to add or delete processing elements and by using passive connections the reliability is improved. The principal disadvantage of this topology is the vulnerability of the network to a failure of the bus



a. Distributed Bus Topology



b. Centrally Controlled Bus Topology

Figure 3.2. Bus Topology

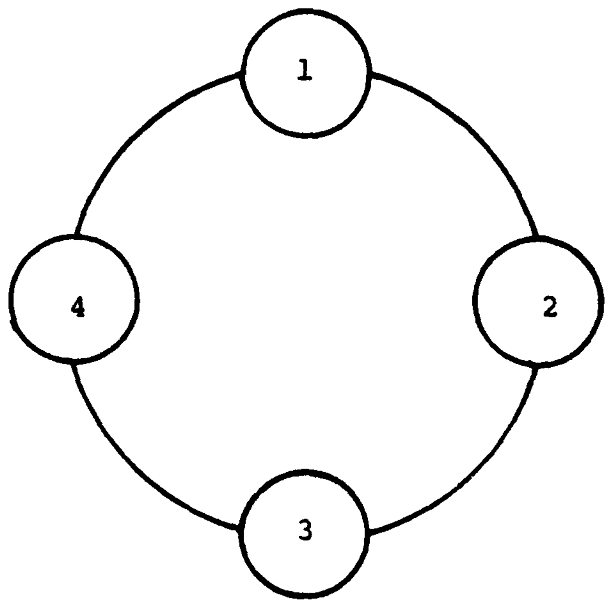
itself. However, nodal failures will generally have no impact on the operation of other nodes.

2. Ring Topology

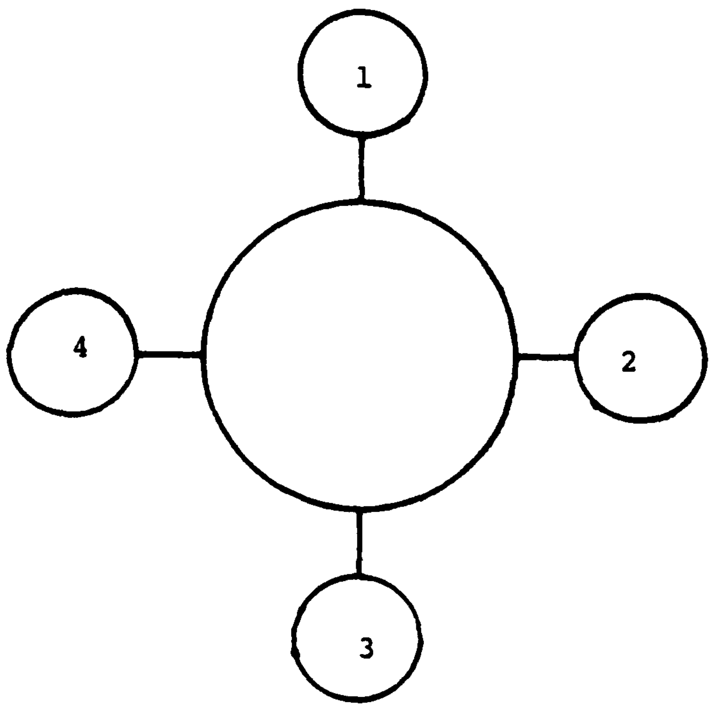
A network topology is characterized as ring, if a collection of processing elements (*computers, peripherals*) are interconnected via a communication path in the form of a loop. Two varieties of loop, or ring, topology are shown in Figures 3.3a and 3.3b [Ref. 14]. In Figure 3.3a, each node acts as a message store-and-forward node. In Figure 3.3b, each node receives bits only in assigned time slots. If all nodes of the loop are of the same type, a distributed ring topology results, but if one node is a loop supervisor then a centrally controlled ring topology results. In general, traffic on the loop flows in one direction only and so a break in the loop at any point disables it. Consequently, each connection to the network is complex because hardware is required to keep the network functioning even when a node fails.

3. Star Topology

A star topology is characteristic of many conventional local and long-distance networks. This arrangement connects each station to a central facility responsible for managing all communications. Its structure is shown in Figure 3-4. Both the advantages and disadvantages of this topology arise from this centralization. Communications and resource management can be efficiently handled, but the limitations



a. Serial Store-and-Forward Ring Topology



b. Broadcast Ring Topology

Figure 3.3. Loop or Ring Topology

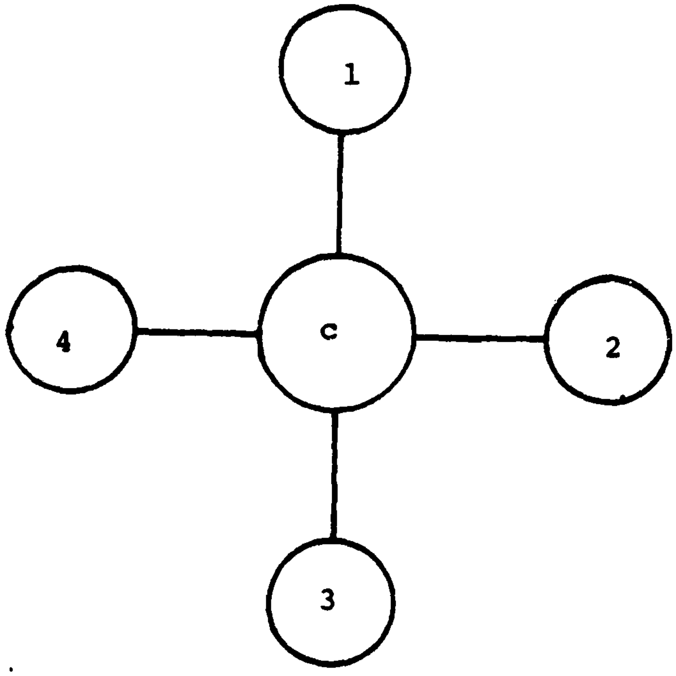


Figure 3.4. Star Topology

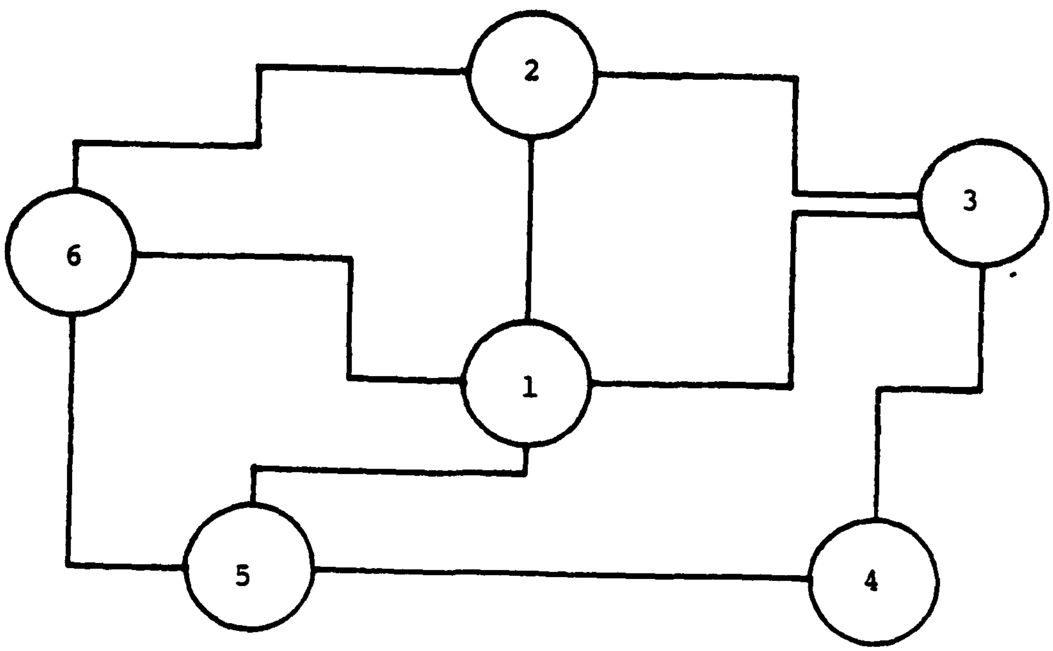


Figure 3.5. Mesh Topology

and reliability of the central unit determine all the network performance characteristics.

4. Mesh Topology

A mesh topology is widely used in long-distance decentralized ("packet-switched") networks. In this arrangement all the nodes are connected arbitrarily as shown in Figure 3.5. The star and the mesh topologies frequently appear in combination in long-distance networks. That is, a mesh network may serve as the hub of a subsidiary star network. By providing alternative routings, the Mesh topology has relatively high reliability, and it allows for unconstrained network reconfiguration.

C. TRANSMISSION MEDIUM

Though the topology selected determines many network issues, it is basically independent of the choice of the transmission medium. The transmission medium provides the physical data communication links of the network, i.e., high speed paths for electrical transmission between two or more nodes or terminals. Generally, local networks can use an assortment of media which are technologically appropriate to a given application.

The four most frequently used transmission media in LAN are: twisted wire-pair, fiber-optics, baseband coaxial cable and broadband coaxial cable (TV-cable).

Twisted wire-pair is basically used to connect stations in a low-speed network where the required bandwidth does not exceed 19 Kbits per second [Ref. 15].

Fiber-optic cable allows the construction of very high performance and secure networks. The advantages of fiber-optic are many, and its potential is very promising. Generally it can provide [Ref. 16]:

1. High-speed data transmission capability (up to one gigabits per second).

2. Relative immunity to electromagnetic and radio frequency interferences.

3. Higher degree of communication security.

4. Large reduction of the bit error rate (one bit per billion). With current technology the fiber-optic cable is not a practical medium for bus networks because of the lack of cable taps.

Most local computer networks use coaxial cable because it allows very high data rates over distances up to several miles without signal regeneration. A coaxial cable is used to support either a baseband channel or a broadband channel. In the first case the coaxial cable provides a single channel i.e., digital signals are placed directly on the cable. In the second, the cable's bandwidth is divided into a number of independent channels by employing Frequency Division Multiplexing (FDM) techniques.

D. PERFORMANCE MEASURES

A basic understanding of computer network performance measurement is essential during the computer network analysis cycle, i.e., from problem definition to simulation experimentation. However, the measurement of computer network performance is difficult and sometimes totally qualitative in nature. This happens due to the great number of different components (hardware and software) that are included in a computer network.

In order to evaluate a network effectively, a set of performance measures must be employed. This set must encompass the network topology, communication devices, transmission facilities, and transmission management software/hardware and treat them as a single system.

The National Bureau of Standards (NBS) [Ref. 17] defines nine measures for evaluating computer network system performance. They are:

1. Availability
2. Transfer Rate of Information Bits (TRIB)
3. Reliability
4. Accuracy (or Residual Error Rate--RER)
5. Channel-Establishment Time
6. Network Delay
7. Channel-Turnaround Delay
8. Transparency
9. Network Security

These nine measures do not represent all the possible performance measures, but they are considered to be the most essential and can be applied to any computer network and provide a basis for comparison with other networks. In a local area network, a computer network manager might be interested also in the utilization rates of the major network components, the throughput, the various network queue lengths, the response time, the mean transmission delay time, etc.

All of the nine major and as many as are necessary of the numerous minor network performance measures should be considered throughout the acquisition process, from network specification to network operation. The degree and the way in which these measures can be considered, varies during the process of going from a conceptual design to an actual implementation in hardware and software. This variance is due to the fact that some performance measures can be applied to a network while it is still in the design phase with a greater accuracy than some of the other performance measures, because some measures require an actual selection of hardware or software before they can be considered accurately. With regard to the numerous minor performance measures that can be considered, their use is dependent basically on the desires and needs of the network analyst or designer.

With respect to a LAN for the implementation of SPLICE functions, some particular performance measures are of great importance during the first stages of network design. At these stages, in fact, a designer primarily needs to estimate message propagation delays and sustained point to point throughput rates. Based on this need this thesis in Chapter IV will discuss the specifications of a simulation model for evaluating these performance measures. Also, after adopting a bus topology for the SPLICE LAN [Refs. 18,19], performance measures such as acquisition probability, waiting time and efficiency of the channel [Ref. 11], can be considered.

The above mentioned performance measures are described as follows:

1. Availability

Availability is defined in Reference (20) as "the portion of a selected time interval during which the information path is capable of performing its assigned data communication function." Alternatively, availability can be defined in terms of the ratio MTTF/(MTTF+MTTR), where MTTF is the mean time to failure and MTTR is the mean time to repair a device. Thus, availability can be improved by increasing MTTF and decreasing MTTR.

Availability of a computer network is decreased not only by component failures but also by transmission overloads caused by either messages which contend for a transmission channel or by the inability of the transmission processing equipment to handle all the transmission requests.

One common problem in a data communication system is to distinguish between very short term failures of a system such as those that cause a series of bit errors and longer term failures that make a system unavailable [Ref. 16]. The distinction is arbitrary and is left to the user to define it in the most meaningful terms. Generally, events that cause errors on communication lines are not defined as failures unless the facility is unusable for more than several seconds or minutes. "Unusable" may mean totally unavailable or having such a high error rate that the effective transfer rate (TRIB) is reduced below some minimum acceptable value.

2. Transfer Rate of Information Bits (TRIB)

Transfer rate is defined as the ratio of the number of information bits accepted by a receiving terminal during a single information transfer period to the duration of the transfer period. Transfer rate is expressed in bits per second.

In calculating network transfer rate, the transmission channel capacity which is specified by the medium is the upper limit for the transfer rate. In an operating network, the transmission management procedures, propagation delay, channel turnaround time, and the retransmission of erroneous messages all subtract from the upper limit of this transfer rate. It should be noted that there are different definitions of what exactly constitutes an information transfer. Most communication engineers consider transmission block headers and trailers to be part of the useful information transferred. On the other hand, some applications-oriented users consider such headers and trailers as an overhead factor and not part of the useful information transferred.

3. Reliability

Reliability is generally defined as the probability that a device will perform without failure for a specified time period or amount of usage. It is important to network users because it gives the probability that a requested task, using networking facilities, has been successfully completed. Expressed as a percentage, reliability differs from availability in that it describes the performance of a network after it has accepted a request from its source.

4. Accuracy

Accuracy or Residual Error Rate (RER) is defined as the ratio of undetected error bits received at a terminal station to the number of information bits transmitted to that terminal station. The value of the residual error rate is computed by the equation

$$RER = (C_e + C_u + C_d)/C_t$$

where:

C_e = the number of erroneous information bits accepted by the receiving terminal station.

C_u = the number of information bits transmitted, but not received by the terminal station.

C_d = the number of duplicate information bits accepted by the receiving terminal station, though they were not intended for duplication.

C_t = the number of information bits in the total transmission.

Accuracy is expressed in the number of error bits per the number of information bits transmitted.

5. Channel-Establishment Time

Channel-establishment time is defined as the period of time required for network communication devices, transmission medium, and transmission management software/hardware to connect a calling terminal station with a called terminal station. The channel-establishment time includes both the time required to place the transmission request and the time required for the network to complete the connection. Channel-establishment time is normally expressed in terms of seconds for most networks but some newer networks are designed for establishment times of not more than several hundred milliseconds.

In networks employing transmission techniques such as packet switching, the conventional interpretation of channel-establishment time does not apply. One view of such networks is that there is no establishment time, since the network can almost always accept the next input message from the sending station with no delay. Another view is that this delay is imposed on every data exchange between stations, since virtual channels must be established.

The significance of a channel-establishment time varies with the network user applications. If a user expects to have many dialogs he may find channel-establishment time to be very important.

6. Network Delay

Network delay or message transfer time is defined as the period of time required for a message to be transmitted from a source terminal station to a receiving terminal station. Network delay is usually expressed in terms of milliseconds. In long-distance networks the actual value of delay depends on physical distances.

In general, the network delay depends on the characteristics of the terminal facilities, i.e., their buffer capacities, the volume of information being transferred, the protocol required, and the reliability of the transmission medium.

In an interactive environment involving a series of short conversational information exchanges, long network delays are undesirable. This is because the data input process has to be slowed down, becoming thus inconvenient and annoying to the user.

7. Channel-Turnaround Delay

Channel-turnaround delay is defined as the time required by a half-duplex transmission channel to reverse its direction of transmission. This performance measure generally depends on the type of channel facilities which are used and it can range from very small values up to several hundred milliseconds.

Channel-turnaround delay reduces the information transfer rate for half-duplex channels by increasing the time required for a block of data acknowledgements. As with network delay, this delay can be reduced to some degree by using longer message blocks and larger storage buffers.

8. Transparency

Transparency is a totally qualitative term which describes the lack of code or procedural constraints which are imposed on network information processing by the network communication devices, transmission facilities, and transmission management software/hardware.

The lack of transparency creates a need for changes in the network components or a need for user education in order to avoid conflict between the information processing and the communication portions of the computer network. Most communication protocols have a problem with code transparency. For example, some protocol may not allow an end-of-transmission code to be used within the text of a message. In this case, the entire message must be scanned and modified in order to remove any illegal bit sequences. This detection and modification process, which may be handled in hardware or software, extends the length of the message and reduces the information transfer rate.

9. Network Security

Network security is another totally qualitative term used to describe the degree of protection provided for the information handled in a computer network from unauthorized access. The importance of network security, of course, depends on the sensitivity of the information being handled. For most military and governmental networks, security is very important.

The security of information is affected by the processing and communication devices, the transmission facilities, the transmission management software/hardware, the terminal station, the authorization that the users of a terminal station have to use it, and the isolation of user's files from each other. Of these network components, the communication devices, the transmission facilities, and the transmission management software/hardware are the most vulnerable to the unauthorized access of information.

The most effective method of providing information security is through information encryption. Encryption is best applied on an end-to-end basis, because it provides protection over the full length of the message's transmission path.

IV. SPECIFICATIONS OF THE LAN SIMULATION MODEL

A. INTRODUCTION

Martin [Ref. 8] defines a simulation model to be: "a logical-mathematical representation of a concept, system, or operation programmed for solution on a high-speed electronic computer." A simulation model can be constructed and applied during any phase of system design or predesign conceptualization. It depends upon the desired answers about the system to decide when a model has to be constructed. Thus, a simulation model can be constructed [Ref. 8]:

1. Before the system is designed in order to determine parameter sensitivity and to optimize or evaluate the system design.

2. During the system design phase in order to test and experiment with system design concepts.

3. After a system has been designed and built in order to supplement system test results and to evaluate overall system effectiveness.

The SPLICE project at the Naval Postgraduate School, Monterey, California, is now in the predesign phase of a Loca- Area Network (LAN) system. This thesis, as part of the project, intends to support the design of a LAN implementing the SPLIC functions, by discussing in this chapter the specifications of a simulation model which will help in parameter sensitivity analysis and LAN design optimization and evaluation.

1. Background

Simulation models have been widely used to study performance of computer systems. In the area of computer networks and especially of LANs, several simulation projects have been reported recently [Refs. 21, 22, 23] but they tend to place emphasis on the development and verification of the communication and access protocols.

With respect to a LAN employing a bus architecture, as the one described in the SPLICE Request for Proposal [Ref. 2] for the implementation of SPLICE functions, there has been little work done on the modeling of such a network. That is, to model the communication system as well as the functions of the nodes. Tropper [Ref. 21] suggests that: "A potentially effective approach to the modeling of such a system would be to use Jackson's open queueing networks to model the nodes, and to use the appropriate time-delay formula to model the performance of the bus itself."

2. Overview

Following the approach proposed by Tropper this chapter will describe a Local Area Network (LAN) simulation model as an open queueing network. First, the simulation model resources are described in terms of the components which compose the model LAN, i.e., the node processor, the node-to-local network adaptor (LNA) interface, the LNA communications software/hardware, and the transmission medium. Next, how these resources are modeled as an open network of queues is described. Then, the model workload is defined assuming an on-line environment with different classes of transactions. Finally, the transaction flow through the LAN simulation model and the selection of transaction classes are described in terms of the stochastic processes representing the flow through the queueing network.

B. SIMULATION MODEL RESOURCES

The simulation model resources are described below in terms of the components which compose the LAN model depicted in Figure 4.1. For that particular design the resources which serve user transactions and contribute to delay are the following:

1. Node Processor,
2. Node-to-Local Network Adaptor (LNA) Interface,
3. LNA Communications Software/Hardware,
4. Transmission medium.

The resources of the model are characterized by their capacity to service requests, where, the request is characterized by the resource capacity it consumes. A description of each resource and its characteristics follows.

1. Node Processor

A node processor in the LAN configuration of Figure 4.1 can be a Host or Front-End Processor (FEP) or a processor implementing one or more SPLICE functions (e.g., Terminal

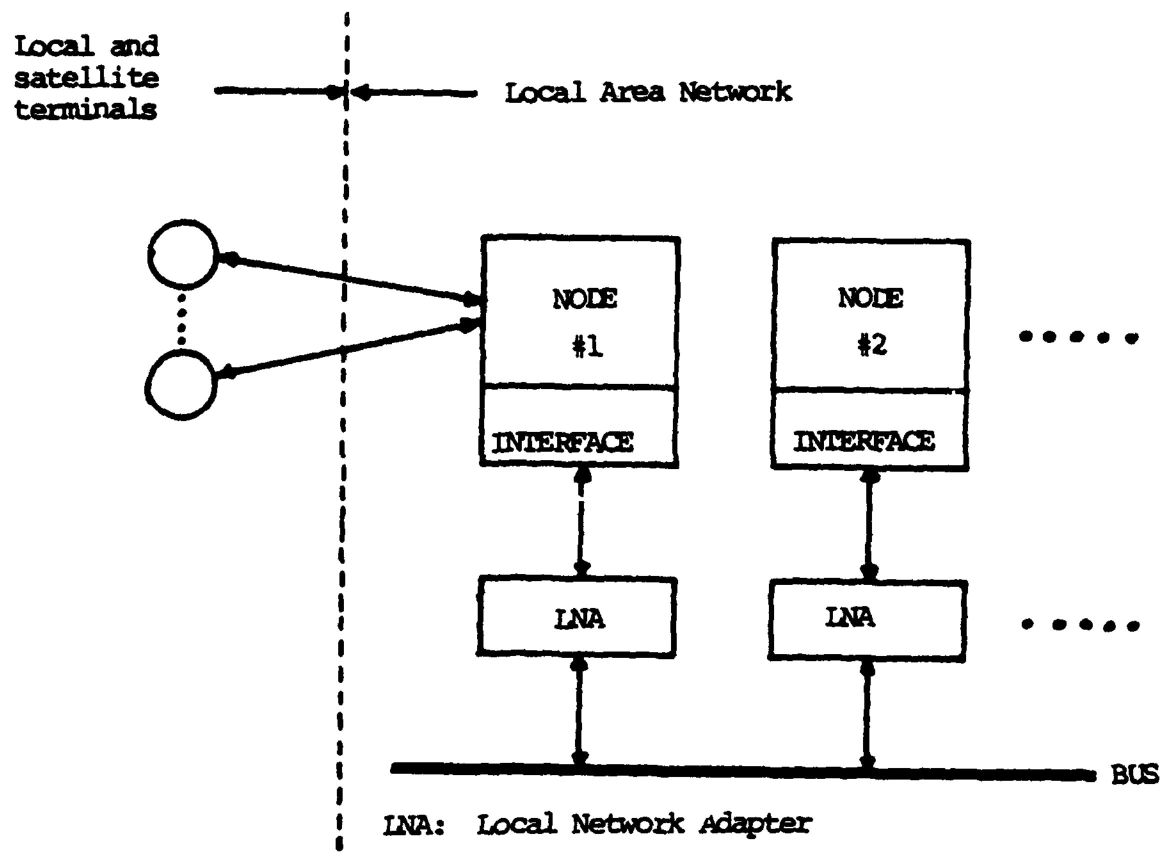


Figure 4.1. LAN Model Components

Mgt., Data Base Mgt., etc. [Refs. 1, 19]. Each node contributes to response delay by having to process applications and execute high-level protocols residing in the node. A typical application in a LAN environment with a delay requirement is a query-response activity in which a user at a terminal connected to a node (FEP) inputs a query to a data base that resides on another node. Using the term "application" in general, different application processes reside in different nodes of the LAN along with nodal communications software.

The execution of applications software residing in the nodes is initiated by requests from users at terminals or processes in other nodes. The resources which are consumed in the execution of the applications and nodal communications software can be characterized by the following factors representing design parameters of the LAN:

a. Number of instructions executed (path length),

b. Node processing capacity,

c. Workload mix.

The workload mix represents the concurrent tasks which are executed in addition to the task being executed [Ref. 24]. The above factors, as well as factors characterizing the other model resources discussed later, will be used as user input values during the implementation phase of the LAN simulation model.

2. <u>Node-to-LNA Interface</u>

The node-to-LNA interface is a physical serial half-duplex server. However, logically, it can support full duplex data transfer. It can be characterized by delay components with:

a. Waiting time,

b. Transmission time,

c. Propagation time.

The propagation time is usually negligible since the distance between the nodes and LNA is typically short. The waiting plus the transmission time is a function of:

a. the total traffic for each interface, and

b. the transfer rate of the interface.

3. LNA Communications Software/Hardware

The LNA is assumed to be a microcomputer-based component which implements the data link layer. The protocol of this layer which resides in the LNA can be implemented in software or hardware. For software implementation, the LNA is characterized by the same factors discussed earlier in applications and nodal communications software. For the hardware implementations, the associated delay is characterized by the hardware service rate.

4. Transmission Medium

The transmission medium serves as the physical communication path in the LAN. The delay associated with it is characterized by:

a. the data transfer rate of the transmission media,

b. the overhead imposed by the access protocols,

c. the distance between nodes.

As it was stated earlier, all the characteristics of the LAN's model resources which can be considered as design parameters can be used as user input values during a simulation run in order to evaluate the sensitivity of different performance measures.

C. LAN RESOURCES MODELING

This section describes how the LAN resources specified earlier are modeled as an open network of queues. Different kinds of processing activities performed by the LAN servers (i.e., the node processor, the node-to-LNA interface, the LNA processor, and the transmission medium), represent the delay components of the network.

The queueing network which represents the LAN system is decomposed into modules corresponding to the network servers. For each module the classes of activities with their priority, the assumptions for the module, the delay equations, and an overhead factor are given below. The models for each of the four modules are specified as follows:

1. Node Processor Modeling

The node processor can be modeled as a single resource multi-class queueing system with a preemptive-resume priority service discipline, as depicted in Figure 4.2. Concerning a SPLICE LAN configuration a node can be a Front-End Processor (FEP), a Host computer, or a Mini-computer implementing SPLICE functions. Each of these nodes includes Central Processor (CP) and dedicated resources (e.g., Memory,

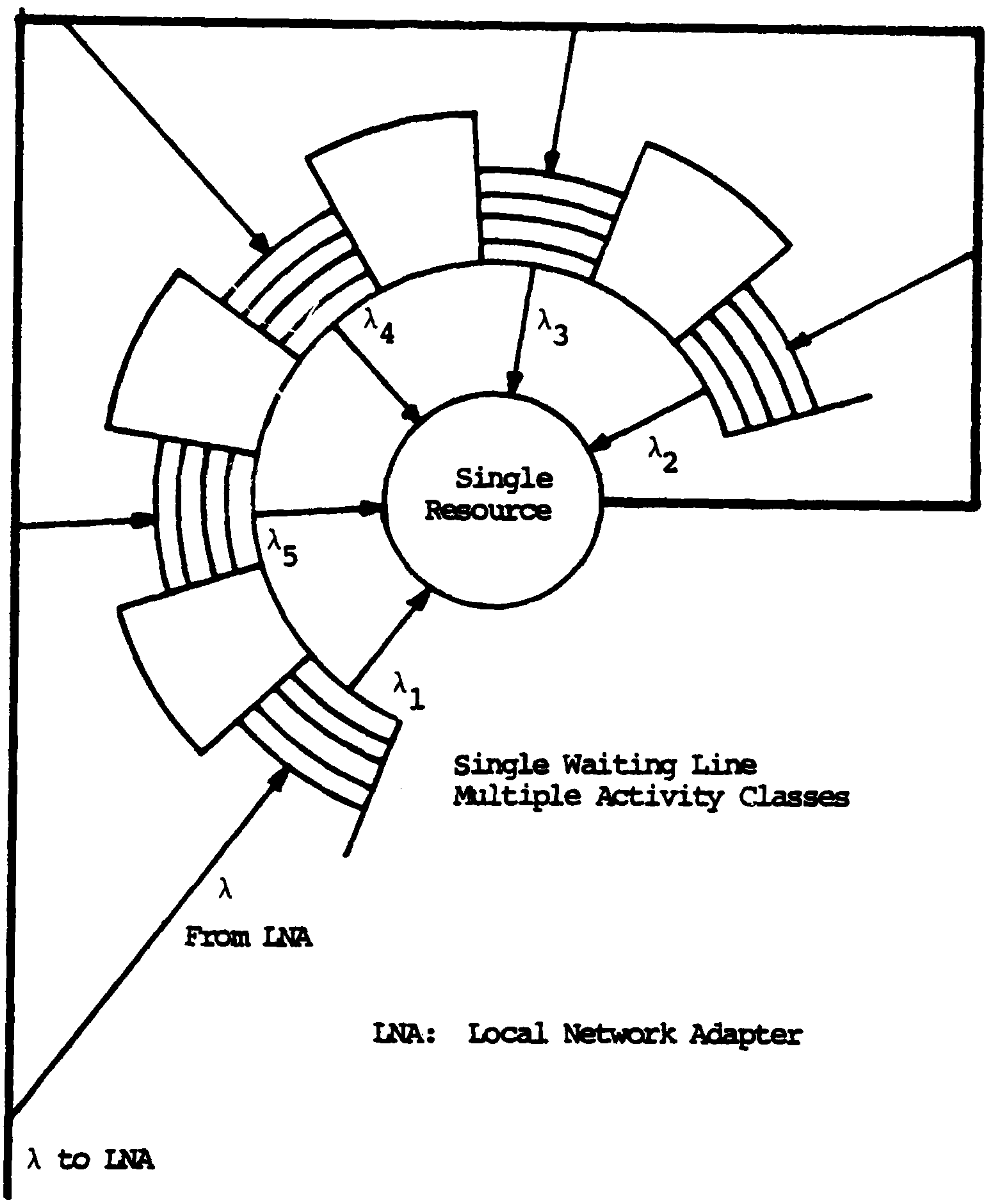


Figure 4.2. Node Queueing System

Disks). The delay imposed by the Disk access requirements must be included during a simulation run for calculating message delay and response time of any transaction processing. Also Disks depend upon the processor to handle interrupts for Disk I/O. This demand can be specified in the model as overhead (NOH: Node Overhead) to the node processor. This overhead, which also includes other LAN administrative functions, affects the nominal capacity of the node processor. The Node Effective Processing Capacity (NEPC) is given [Ref. 24] by:

$$NEPC = (1 - NOH)NIPS,$$

where NIPS (Node Instructions Per Second) is the average number of machine instructions per second that the node processor can execute.

A variety of activities can be considered for each node. In the LAN simulation model five classes of activities are specified which are serviced on a preemptive-resume scheme [Ref. 24]. These activities are:

a. Link-In: Link level functions for messages inbound from the local LNA.

b. Link-Out: Link level functions for messages outbound from the node to the local LNA.

c. Protocols-In: High level protocol functions for messages inbound from the local LNA.

d. Protocols-Out: High level protocol functions for messages outbound to the local LNA.

e. Application: Transaction segment processing.

The service time for each activity can be characterized by the activity path length in machine instructions and the effective processor capacity (NEPC) of the node processor.

A priority structure for servicing the five classes of activities has to be specified for every node in the LAN (FEP, Host, etc.). For example, the priority structure for a FEP node is assumed to be (from highest to lowest):

a. Protocols-Out

b. Link-Out

c. Application

d. Protocols-In

e. Link-In.

This priority structure is independent of any other priority scheme based upon the type of transactions and which will affect only the queue discipline of class of activities. The probable transaction priority is not considered here to avoid unnecessary complexity of the model.

Assumptions for the node queueing model are as follows:

a. The queueing system has a single waiting line and a single server.

b. Each class of activity is represented by an independent Poisson process with mean arrival rate λ_i.

c. The service times of each class of activity is represented by an exponential distribution with mean service time S_i.

The rationale for choosing these assumptions concerning arrival of classes of activities and service times is well documented [Refs. 25, 26, 27, 29].

The equations for node delay, T_{gj}, for the various classes of activities are given [Ref. 25] by:

$$T_{gj} = \sum_{i=1}^{j} \lambda_i S_i / (1 - \sum_{i=1}^{j} \rho_i) + S_j \times 1/(1 - \sum_{i=1}^{j-1} \rho_i)$$

where for a node g the number of activity classes is j and the server utilization $\rho_i = \lambda_i S_i$. The λ_i's are determined by solving the balance equations for the network of queues associated with the node module.

2. Node-to-LNA Interface Modeling

The node-to-LNA interface can be modeled as a single server queue. An overhead factor (IOH: Interface OverHead) can be considered on the nominal transfer rate of the interface for non-data bits. Thus, the Effective Transfer Rate (ETR) for the interface is given by:

$$ETR = (1 - IOH)\ ITR,$$

where ITR is the nominal Interface Transfer Rate in bits per second (bps).

Assumptions for the interface queueing model are the following:

a. The queueing system has a single waiting line and single server.

b. There are two classes of arrivals (messages from the node to the LAN and messages from the LNA to the node) without priority structure (i.e., the service discipline is First-Come, First-Served).

c. Each arrival class is represented by an independent Poisson process with parameter λ_i.

d. The service time of each arrival class is represented by an exponential distribution with mean service time S_i.

The equations for the interface delay of node g with j arrival classes, T_{gj}, are given [Ref. 24] by:

$$T_{gj} = \rho\overline{S}/(1 - \rho) + S_j,$$

where

$$\rho = \lambda'\overline{S}$$

$$\lambda' = \lambda_1 + \lambda_2$$

$$\overline{S} = (\lambda_1 S_1 + \lambda_2 S_2)\lambda'$$

3. LNA Processor Modeling

The LNA processor can be modeled as a single resource, multi-class queueing system with a preemptive-resume priority structure as the one given in Figure 4.2. An overhead factor (AOH: Adaptor OverHead) can be considered on the processing capacity of the LNA processor, which includes system's administration overhead functions. Thus, the effective processing capacity (AEPC: Adaptor Effective Processing Capacity) for a LNA processor is given by:

$$AEPC = (1 - AOH)\ AIPS,$$

where AIPS is the average number of machine instructions per second that the LNA processor can execute.

For the LNA module in the simulation model five classes of activities can be specified. A list of these activities follows [Ref. 24]:

a. Network Link-In: Link level functions for messages inbound to the LNA from the network transmission medium.

b. Network Link-Out: Link level functions for messages outbound from the LNA to the network.

c. Node Link-In: Link level functions for messages inbound to the LNA from the node side.

d. Node Link-Out: Link level functions for messages outbound from the LNA to the node.

e. Protocol: Execution of data link protocol for messages from the network side and the node side. There is a FCFS discipline within this class of activity.

The priority structure of classes of activities for a LNA processor is assumed to be (from highest to lowest):

a. Network Link-Out

b. Node Link-Out

c. Network Link-In

d. Node Link-In

e. Protocol

Again the priority structure is independent of any other priority scheme which can be imposed at transaction level.

The mathematical assumptions as well as the equations for LNA model are the same as those given earlier for the node model.

4. Transmission Medium Modeling

The combination of the physical protocol layer of the OSI model and the transmission medium can be modeled as a single server queue. The physical protocol layer is implemented in hardware and is referred to as the Media Access Unit (MAU). The transmission medium access scheme considered in the simulation model is the Carrier Sense Multiple Access with Collision Detection (CSMA/CD) which has been modeled extensively [Ref. 21].

The assumptions for the queueing analysis of the CSMA/CD access scheme are [Ref. 21]:

a. Poisson arrivals over an infinite population.

b. Negligible collision detection time (compared to the bus propagation time).

c. Channel sensing is instantaneous.

d. Propagation time between any two nodes is uniform and equal to the maximum value.

e. Retransmission of packets is ccording to the Binary Exponential Backoff probability rule.

f. The random interval parameters for the backoff algorithm is uniformly distributed and the same for all MAU's.

A priority message system can be supported at the transmission medium level by varying each MAU's backoff algorithm (assumptions "e" and "f" above) and letting some of them have linear instead of exponential backoff algorithm [Ref. 22].

An algorithm for the CSMA/CD access scheme is given [Ref. 24] in Figure 4.3.

The equation for the time delay, D, in terms of the system throughput, the offered load, and other system parameters is given by the formula [Ref. 26]:

$$D = (A1 + A2 + A3)T$$

where

A1 = the normalized wasted time due to collisions,

A2 = the dead time due to retransmissions and rescheduling,

A3 = the propagation and transmission time, and

T = the packet length.

Given that:

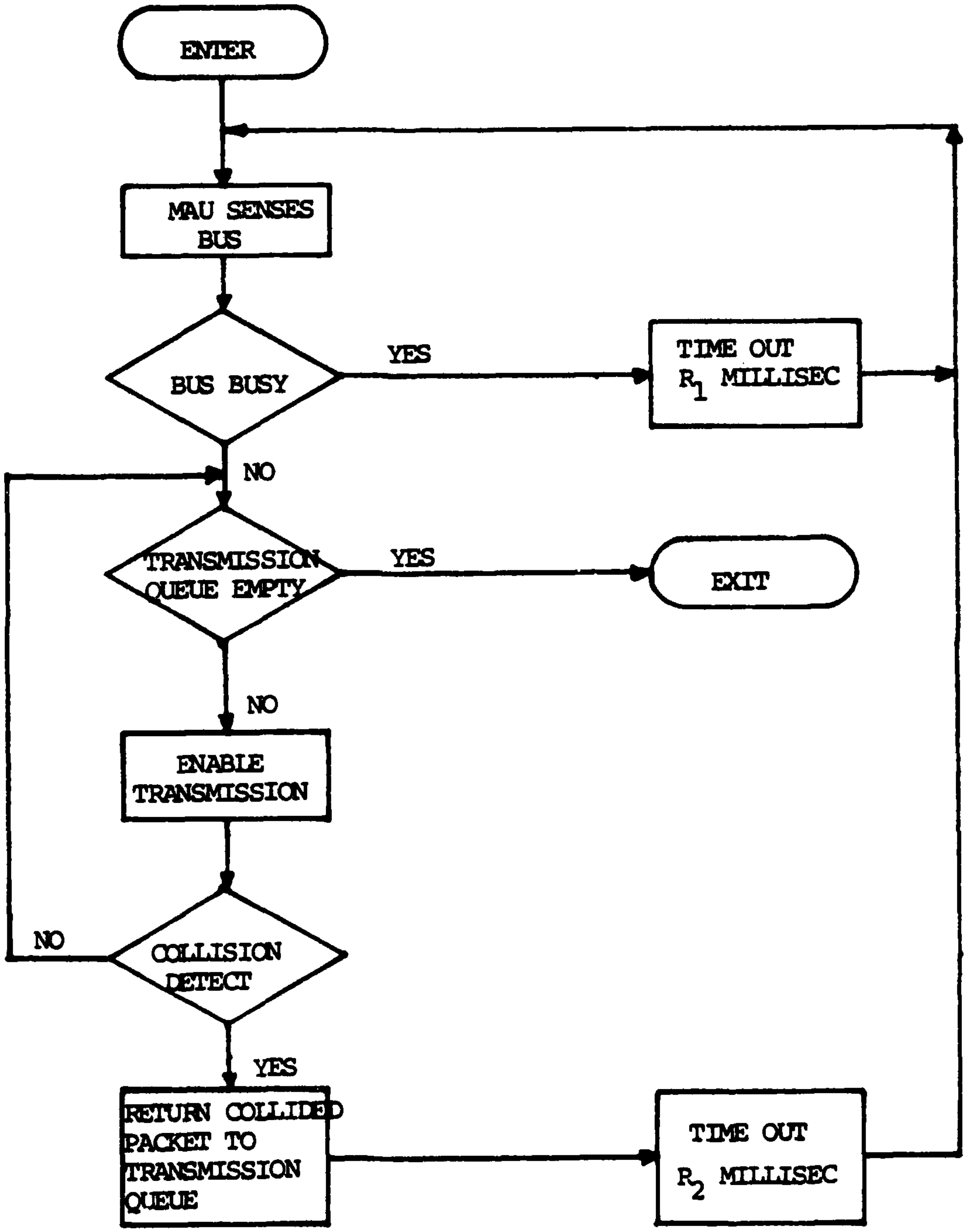


Figure 4.3. CSMA/CD Algorithm

N1 = the average number of times a packet encounters a collision or busy state,

N2 = the average number of times a packet encounters a collision,

R1 = the normalized mean retrial interval after detecting a busy condition,

R2 = the normalized mean retrial interval after detecting a collision,

a = the progagation delay,

G = the offered channel traffic (load), and

S = the throughput.

It follows that:

$$A1 = N2(W + a)$$

$$A2 = R2(2^{N_2+1} - 1) + (N1 - N2)R1$$

$$A3 = a + 1$$

where:

$$W = (1 - e^{-aG})/G - ae^{-aG}$$

$$N1 = G/S - 1$$

$$N2 = (1 - aG)\ e^{-aG} - 1$$

and the normalized throughout ($S = \lambda T$) is related to the normalized offered packet traffic (including retransmissions), G, by the equation [Ref. 27]:

$$S = \frac{G\,e^{-aG}}{(1+a)\,G\,e^{aG} + (1+aG)\,(1-e^{-aG})^{2} + 1}$$

C. MODEL WORKLOAD

For the particular LAN configuration to be modeled (Figure 4.1), the network user's transactions, when translated into access and processing requirements, can be characterized by the type and the amount of system resources the network will have to provide in order to fulfill these requirements. The sum of system resource requirements generated by all the network users represents the system or network workload. Generally, the workload of a computer system has certain basic statistical characteristics that do not change greatly over short periods of time. The use of such characteristics make it possible to do the following [Ref. 28]:

1. To characterize the system workload by statistical distributions of requirements placed on individual system resources, and

2. To define a unit of work and then express the workload characteristics in relation to this unit.

The workload characterization of a SPLICE LAN configuration would be based on an on-line and batch input environments. The simulation model specified in this chapter considers an on-line environment with twelve different classes of transactions [Ref. 2]. This is because it was felt that to include the batch input environment added detail and complexity to the simulation model which was unnecessary for the fulfillment of analysis objectives during the early stages of SPLICE LAN design.

Each class of transaction requires service from resources of one or more network nodes and is divided into a number of transaction segments (processes). A description of what constitutes a transaction class and how the on-line workload is specified follows.

1. Transaction Class

The transactions input from the terminals at the FEP node of the LAN are described as a series of transaction classes. A transaction can be characterized according to its arrival rate which can be converted into probability of occurrence or cumulative probability of occurrence, and the number of LAN node requests (data path) it requires. For example, suppose the LAN contains five nodes. Table 4.1 shows the twelve transaction classes and their hypothetical arrival rates, probabilities of occurrence, and node requests. For example, transaction class TC2 describes those transactions which require three node requests (i.e., data path used 1-2-5). The data path is determined from the required LAN resources by the segments (processes) of this transaction class. The probability of such a transaction (TC2) to occur in the transaction input stream of node #1 (FEP) is five percent.

TABLE 4.1

On-Line Workload Characterization
(Hypothetical Data)

TRANSACTION CLASS	PEAK HOUR TRANSACTION ARRIVAL RATE (TRANS/SEC)	PROBABILITY OF OCCURRENCE	CUMULATIVE PROBABILITY OF OCCURRENCE	LAN NODE REQUEST (NODE-NUMBER)
TC1	3.07888	0.17	0.17	1-2
TC2	0.88445	0.05	0.22	1-2-5
TC3	0.59364	0.03	0.25	1-3-5
TC4	0.41312	0.02	0.27	1-2-3-5
TC5	1.58868	0.09	0.36	1-3-4
TC6	1.95634	0.11	0.47	1-2-3-4
TC7	0.71295	0.04	0.51	1-3-4-5
TC8	0.84101	0.05	0.56	1-2-3-4-5
TC9	0.28638	0.02	0.58	1-4-5
TC10	1.42839	0.08	0.66	1-2-4-5
TC11	0.60543	0.03	0.69	1-2-3
TC12	5.64292	0.31	1.00	1-2-4

2. Transaction Segment

Each class of transaction is composed of a series of transaction segments. These segments are small units of work within a transaction which compete for LAN resources. Reference (2) recognizes eight types of transaction segments, the following:

a. Edit

b. Validation read

c. Validation

d. Error message

e. Processing read

f. Processing

g. File write

h. Format output.

The process load for each transaction class along with their segments characteristics and corresponding hypothetical values are given in Tables 4.2 through 4.13. It can be seen that a transaction class is divided into four to eight segments with different message length, characteristics, and processing requirements for each.

E. TRANSACTION FLOW

A transaction flow through the LAN system simulation model is a flow through the network of queues, each modeling one resource of the system model. The queueing network which represents the LAN system, as it was discussed earlier, is not considered and modeled as a whole, but it is decomposed into modules which are analyzed in isolation. The focus in this approach is on the stochastic processes representing the flow through each module where the output of one module represents the input to a subsequent module.

The flow in the simulation model can be portrayed as a series of discrete events. The occurrence or timing of these events is on a next event scheduled basis. The next event approach to simulation is discussed briefly by Fishman [Ref. 6]. Also the occurrence of events is governed according to the various statistical distributions of requirements which

TABLE 4.2

Process Load for Transaction Class 1

TRANSACTION SEGMENT	CHARACTERISTICS	VALUE
1. Edit	a. Message length	2
	b. No. of instructions	40
	c. % of fail	1
2. Validation read	a. No. of records	1
	b. Record length	1500
	c. No. of instructions per access	5
	d. % of fail	0
3. Validation	a. No. of instructions	0
	b. % of fail	0
4. Error Msg	a. No. of instructions	5
	b. Message length	80
5. Processing read	a. No. of records	0
	b. Record length	0
	c. No. of instructions	0
6. Processing	a. No. of instructions	0
7. File write	a. No. of instructions	0
	b. No. of modified record	0
	c. Length of modified record	0
	d. No. of adds	0
	e. Length of added record	0
	f. No. of indices	0
8. Format output	a. No. of instructions	5
	b. Message Length	1500
	c. No. of records	1

TABLE 4.3

Process Load for Transaction Class 2

TRANSACTION SEGMENT		CHARACTERISTICS	VALUE
1. Edit	a.	Message length	200
	b.	No. of instructions	50
	c.	% of fail	1
2. Validation read	a.	No. of records	10
	b.	Record length	250
	c.	No. of instructions per access	20
	d.	% of fail	1
3. Validation	a.	No. of instructions	0
	b.	% of fail	0
4. Error Msg	a.	No. of instructions	5
	b.	Message length	80
5. Processing read	a.	No. of records	0
	b.	Record length	0
	c.	No. of instructions	0
6. Processing	a.	No. of instructions	0
7. File write	a.	No. of instructions	0
	b.	No. of modified record	0
	c.	Length of modified record	0
	d.	No. of adds	0
	e.	Length of added record	0
	f.	No. of indices	0
8. Format output	a.	No. of instructions	50
	b.	Message length	1000
	c.	No. of records	1

TABLE 4.4

Process Load for Transaction Class 3

TRANSACTION SEGMENT		CHARACTERISTICS		VALUE
1.	Edit	a.	Message length	200
		b.	No. of instructions	50
		c.	% of fail	1
2.	Validation read	a.	No. of records	18
		b.	Record length	350
		c.	No. of instructions per access	20
		d.	% of fail	1
3.	Validation	a.	No. of instructions	0
		b.	% of fail	0
4.	Error Msg	a.	No. of instructions	5
		b.	Message length	80
5.	Processing read	a.	No. of records	0
		b.	Record length	0
		c.	No. of instructions	0
6.	Processing	a.	No. of instructions	0
7.	File write	a.	No. of instructions	0
		b.	No. of modified record	0
		c.	Length of modified record	0
		d.	No. of adds	0
		e.	Length of added record	0
		f.	No. of indices	0
8.	Format output	a.	No. of instructions	50
		b.	Message length	1800
		c.	No. of records	1

TABLE 4.5

Process Load for Transaction Class 4

TRANSACTION SEGMENT	CHARACTERISTICS	VALUE
1. Edit	a. Message length	50
	b. No. of instructions	100
	c. % of fail	1
2. Validation read	a. No. of records	1
	b. Record length	100
	c. No. of instructions per access	10
	d. % of fail	1
3. Validation	a. No. of instructions	50
	b. % of fail	1
4. Error Msg	a. No. of instructions	30
	b. Message length	500
5. Processing read	a. No. of records	0
	b. Record length	0
	c. No. of instructions	0
6. Processing	a. No. of instructions	0
7. File write	a. No. of instructions	20
	b. No. of modified record	0
	c. Length of modified record	0
	d. No. of adds	1
	e. Length of added record	100
	f. No. of indices	5
8. Format output	a. No. of instructions	20
	b. Message length	500
	c. No. of records	1

TABLE 4.6

Process Load for Transaction Class 5

TRANSACTION SEGMENT	CHARACTERISTICS	VALUE
1. Edit	a. Message length	175
	b. No. of instructions	100
	c. % of fail	5
2. Validation read	a. No. of records	8
	b. Record length	250
	c. No. of instructions per access	20
	d. % of fail	2
3. Validation	a. No. of instructions	150
	b. % of fail	1
4. Error Msg	a. No. of instructions	50
	b. Message length	600
5. Processing read	a. No. of records	5
	b. Record length	200
	c. No. of instructions	10
6. Processing	a. No. of instructions	175
7. File write	a. No. of instructions	30
	b. No. of modified record	5
	c. Length of modified record	200
	d. No. of adds	2
	e. Length of added record	250
	f. No. of indices	10
8. Format output	a. No. of instructions	30
	b. Message length	1000
	c. No. of records	1

TABLE 4.7

Process Load for Transaction Class 6

TRANSACTION SEGMENT	CHARACTERISTICS	VALUE
1. Edit	a. Message length	800
	b. No. of instructions	250
	c. % of fail	8
2. Validation read	a. No. of records	20
	b. Record length	350
	c. No. of instructions per access	30
	d. % of fail	3
3. Validation	a. No. of instructions	500
	b. % of fail	2
4. Error Msg	a. No. of instructions	50
	b. Message length	1500
5. Processing read	a. No. of records	10
	b. Record length	350
	c. No. of instructions	20
6. Processing	a. No. of instructions	250
7. File write	a. No. of instructions	30
	b. No. of modified record	15
	c. Length of modified record	250
	d. No. of adds	15
	e. Length of added record	350
	f. No. of indices	10
8. Format output	a. No. of instructions	50
	b. Message length	1500
	c. No. of records	1

TABLE 4.8

Process Load for Transaction Class 7

TRANSACTION SEGMENT	CHARACTERISTICS	VALUE
1. Edit	a. Message length	175
	b. No. of instructions	100
	c. % of fail	1
2. Validation read	a. No. of records	1
	b. Record length	200
	c. No. of instructions per access	10
	d. % of fail	1
3. Validation	a. No. of instructions	50
	b. % of fail	1
4. Error Msg	a. No. of instructions	30
	b. Message length	500
5. Processing read	a. No. of records	0
	b. Record length	0
	c. No. of instructions	0
6. Processing	a. No. of instructions	0
7. File write	a. No. of instructions	20
	b. No. of modified record	1
	c. Length of modified record	200
	d. No. of adds	0
	e. Length of added record	0
	f. No. of indices	5
8. Format output	a. No. of instructions	20
	b. Message length	500
	c. No. of records	1

TABLE 4.9

Process Load for Transaction Class 8

TRANSACTION SEGMENT		CHARACTERISTICS		VALUE
1.	Edit	a.	Message length	175
		b.	No. of instructions	100
		c.	% of fail	5
2.	Validation read	a.	No. of records	8
		b.	Record length	250
		c.	No. of instructions per access	10
		d.	% of fail	2
3.	Validation	a.	No. of instructions	130
		b.	% of fail	2
4.	Error Msg	a.	No. of instructions	40
		b.	Message length	800
5.	Processing read	a.	No. of records	4
		b.	Record length	200
		c.	No. of instructions	15
6.	Processing	a.	No. of instructions	175
7.	File write	a.	No. of instructions	20
		b.	No. of modified record	10
		c.	Length of modified record	250
		d.	No. of adds	0
		e.	Length of added record	0
		f.	No. of indices	0
8.	Format output	a.	No. of instructions	30
		b.	Message length	750
		c.	No. of records	1

TABLE 4.10

Process Load for Transaction Class 9

TRANSACTION SEGMENT	CHARACTERISTICS	VALUE
1. Edit	a. Message length	30
	b. No. of instructions	300
	c. % of fail	2
2. Validation read	a. No. of records	5
	b. Record length	150
	c. No. of instructions per access	20
	d. % of fail	3
3. Validation	a. No. of instructions	300
	b. % of fail	2
4. Error Msg	a. No. of instructions	100
	b. Message length	80
5. Processing read	a. No. of records	100
	b. Record length	300
	c. No. of instructions	5
6. Processing	a. No. of instructions	500
7. File write	a. No. of instructions	20
	b. No. of modified record	200
	c. Length of modified record	100
	d. No. of adds	100
	e. Length of added record	200
	f. No. of indices	4
8. Format output	a. No. of instructions	50
	b. Message length	132
	c. No. of records	400

TABLE 4.11

Process Load for Transaction Class 10

TRANSACTION SEGMENT	CHARACTERISTICS	VALUE
1. Edit	a. Message length	800
	b. No. of instructions	300
	c. % of fail	10
2. Validation read	a. No. of records	20
	b. Record length	350
	c. No. of instructions per access	30
	d. % of fail	3
3. Validation	a. No. of instructions	500
	b. % of fail	2
4. Error Msg	a. No. of instructions	50
	b. Message length	1500
5. Processing read	a. No. of records	10
	b. Record length	350
	c. No. of instructions	20
6. Processing	a. No. of instructions	250
7. File write	a. No. of instructions	30
	b. No. of modified record	20
	c. Length of modified record	350
	d. No. of adds	0
	e. Length of added record	0
	f. No. of indices	0
8. Format output	a. No. of instructions	50
	b. Message length	1500
	c. No. of records	1

TABLE 4.12

Process Load for Transaction Class 11

TRANSACTION SEGMENT		CHARACTERISTICS		VALUE
1.	Edit	a.	Message length	80
		b.	No. of instructions	30
		c.	% of fail	2
2.	Validation read	a.	No. of records	5
		b.	Record length	150
		c.	No. of instructions per access	20
		d.	% of fail	3
3.	Validation	a.	No. of instructions	500
		b.	% of fail	2
4.	Error Msg	a.	No. of instructions	100
		b.	Message length	80
5.	Processing read	a.	No. of records	25
		b.	Record length	150
		c.	No. of instructions	15
6.	Processing	a.	No. of instructions	2500
7.	File write	a.	No. of instructions	20
		b.	No. of modified record	5
		c.	Length of modified record	250
		d.	No. of adds	2
		e.	Length of added record	75
		f.	No. of indices	3
8.	Format output	a.	No. of instructions	50
		b.	Message length	80
		c.	No. of records	125

TABLE 4.13

Process Load for Transaction Class 12

TRANSACTION SEGMENT	CHARACTERISTICS	VALUE
1. Edit	a. Message length	80
	b. No. of instructions	10
	c. % of fail	1
2. Validation read	a. No. of records	0
	b. Record length	0
	c. No. of instructions per access	0
	d. % of fail	0
3. Validation	a. No. of instructions	0
	b. % of fail	0
4. Error Msg	a. No. of instructions	35
	b. Message length	150
5. Processing read	a. No. of records	1
	b. Record length	80
	c. No. of instructions	10
6. Processing	a. No. of instructions	50
7. File write	a. No. of instructions	10
	b. No. of modified record	0
	c. Length of modified record	0
	d. No. of adds	1
	e. Length of added record	80
	f. No. of indices	2
8. Format output	a. No. of instructions	20
	b. Message length	80
	c. No. of records	1

are placed on individual system resources. This section will discuss transaction flow within the simulation model in terms of transaction class selection and processing.

1. Transaction Class Selection

The selection and processing of a transaction involves the determination of its class and consequently the transaction segments needed, each with their specific processing requirements.

The arrival of transactions at their initial input into the front-end processor can be characterized by a Poisson process, assuming independent and random inputs. Then it can be shown [Ref. 29] that the transaction interarrival times (τ_n) are exponentially distributed about a mean interarrival time. This distribution is described by

$$P(s) = P(\tau_n \leq S) = 1 - e^{-\lambda S}, \quad S \geq 0$$

where:

$P(S)$ = the probability that a transaction will arrive within time S.

λ = the transaction arrival rate at the particular processor.

Since the transactions are assumed to enter the processor's input queue independently and at random, P(S) must be a random number between 0 and 1. Consequently, 1-P(S) is a random number R. Thus,

$$P(S) = 1 - e^{-\lambda S}$$

or

$$e^{-\lambda S} = 1 - P(S) = R$$

or

$$-\lambda S = \ln R$$

or

$$S = - \ln R/\lambda$$

where S is the time increment added to the present time to schedule the arrival of the next transaction at the Front-End Processor (FEP).

The class of the transaction input at the FEP is determined by referring to their cumulative probability of occurrence (Table 4.1) according to a step probability function. This function allows for the independent random selection of transaction class while maintains their relative probabilities of occurrence.

After a transaction has been assigned a class, the corresponding segments (Tables 4.2 through 4.13) of this transaction begin processing following the predetermined data path specified in Table 4.1 (LAN node requests).

When all of the node requests of a particular transaction have completed processing, the various statistics associated with the transaction and the LAN system model are accumulated and updated respectively and the transaction is output from the model.

V. IMPLEMENTATION OF THE SIMULATION MODEL

The three general phases in the construction of a simulation model are: conceptualization (specifications), implementation, and finally execution of the production runs on the computer with interpretation and evaluation of simulation outcomes. The conceptualization of the model was discussed in the previous chapter, where the specifications of a simulation model for a local area network were given. The implementation phase involves the transformation of conceptual model into a tangible computer simulation that is ready to generate simulation outputs during the final phase of the construction.

It is beyond the scope of this thesis to implement and conduct experiments with the specified model. However, some comments on aspects of the last two phases of model construction is considered essential for an overall presentation and understanding of a simulation model. This chapter will briefly discuss programming of the model in terms of simulation languages available, and will comment on simulation experiments.

A. PROGRAMMING THE MODEL

Programming is one of the major tasks in the implementation phase of model construction. It includes all the procedures required to transform the logical statements and mathematical formulations of the model to some computer language statements. Martin [Ref. 8] recognizes the following procedures to be involved in a programming task:

1. Modularization of the model in a programming context in which the model is subdivided into logical units and subunits.

2. Construction of flow chart diagrams that represent the program flow of the model.

3. Coding of the program for the particular computer.

4. Checking the validity of the program and making necessary revisions.

5. Preparation of input and output formats.

A major step before starting actual programming is the determination of a computer system to be used for implementation of the model. For the particular LAN simulation model, specified in Chapter IV as part of the SPLICE project at the Naval Postgraduate School, it would be implemented using the facilities available at the school, i.e., IBM 3033 computer system and GPSS (General Purpose Systems Simulator) or SIMSCRIPT simulation languages. Simulation languages, as distinguished from general-purpose languages, are problem oriented. Such languages are usually written in a largely computer independent notation for a particular problem area, and contain statements or constructs appropriate for formulating solutions to specific types of problems. Of course general-purpose languages such as FORTRAN can be used in simulation modeling, but the simulation languages designed specifically for the purpose of computer simulation provide certain useful features [Ref. 3]. Among them:

1. Reducing the programming task and providing conceptual guidance.

2. Aiding in defining the classes of entities within the sysem and providing flexibility for change.

3. Describing the relationship of entities to one another and to their common environment.

Various simulation languages are operational at the present time. The most popular are GPSS, a process-oriented language, and SIMSCRIPT, an event-oriented language, both available at NPS's Computer Center. Table 5.1 gives a comparative list of some features of GPSS and SIMSCRIPT as briefly summarized by Tocher [Ref. 30]. Each language has certain characteristics and applications, and each of them can be extremely useful when applied properly. A brief description of each of these two simulation languages is given below:

1. <u>GPSS Simulation Language</u>

General Purpose Systems Simulator (GPSS), was developed originally by G. Gordon at IBM and is one of the most popular discrete-event simulation languages. GPSS is process oriented, containing a supply of flow chart-like blocks [Ref. 31]. It also provides a large variety of autonomously generated measurements about the simulated model. GPSS can also be

TABLE 5.1

Features of GPSS and SIMSCRIPT

	GPSS	SIMSCRIPT
What are the dominant types of activities?	Transaction	Transaction
How is the system represented?	Flow chart	FORTRAN-based
What uniform random number generation technique is used?	Multiplicative	Multiplicative
How many random number generators are available?	Indefinite	1
Is there uniform sampling?	Yes	Yes
Is there any statistical collecting in histogram form?	Yes	No
Can the mean be computed?	Yes	Yes
Can the standard deviation be computed?	Yes	Yes
What is the limit on the number of distributions?	100	None
Can arithmetic tests be performed?	Yes	Yes
Can set inclusion tests be performed?	No	Yes

viewed as a program that employs a language which is designed to describe simulation models of a system. The user constructs a logical model of the system using a block diagram consisting of specific block types in which each block type represents some basic system action.

GPSS elements are blocks, transactions, and equipment. Specific block types have a name, a characteristic symbol, and a block number. Each block has a designated block time that indicates the number of time units required for action represented by the block. The block time is not constant; it may vary in a random or nonrandom manner. Transactions are basic units that move through the system. Equipment elements contain facilities and stores. Facilities can handle one transaction at a time, whereas stores can handle many transactions simultaneously.

Since its original version, GPSS has appeared in subsequent, more powerful versions: GPSS-II, -III, -IV, -V, and -360. Current versions provide limited capability for using FORTRAN subroutines. Also there is a version which through a CRT display unit, provides conversational features, user interactive input, and control.

2. SIMSCRIPT Simulation Language

SIMSCRIPT was developed by H. Markowitz, G. Hausner, and H. Karr at the RAND Corporation. It was one of the original discrete-event simulation languages, i.e., is based upon the notion that every model system is composed of elements with numerical values that are subject to periodic change. The state of a system is described in terms of entities, attributes, and sets. The status of a system is changed at discrete points in simulated time by the occurrence of an event. Entities are objects which compose the system, and may be classified as temporary or permanent during the simulation. Attributes are properties associated with entities or items that describe entities. Sets are groups of entities. Status is the numerical description of the simulated system. Events are series of statements grouped together which modify the status of the simulated system at various points in simulated time. There are two types of events possible: exogenous (from outside the simulation system), and endogenous (from within the simulation system). The occurrence of these events is governed by a SIMSCRIPT provided timing routine. This timing routine automatically keeps track of simulated time and causes the various events to occur as they are scheduled by the simulation program. The different kinds of events are enumerated in an events list and a separate event subroutine has to be written for each event. By contrast with GPSS, SIMSCRIPT does not provide for any automatic statistical analysis.

B. SIMULATION EXPERIMENTS

The final phase in the development of the local area network simulation model is to conduct a series of simulation experiments which will be helpful in drawing conclusions about the system modeled. That is, for the particular SPLICE LAN design, potential bottlenecks and other design weaknesses can be identified as well as determining the adequacy of the components of the design relative to specific delay requirements.

In general, simulation experiments are conducted for a wide variety of purposes, some of which are the following [Ref. 3]:

1. Evaluation: determining if a proposed system design performs well in an absolute sense when evaluated against specific criteria.

2. Comparison: comparing competitive systems designed to carry out a specified function, or comparing several proposed operating policies or procedures.

3. Prediction: estimating the performance of the system under some projected set of conditions.

4. Sensitivity analysis: determining which are the most significant factors affecting overall system performance.

5. Optimization: determining exactly which combination of factor levels will produce the best overall response of the system.

In the experimentation phase of model construction system performance criteria must be established that will be used in rating various alternative local area network designs. A frequently employed performance measure in most local area network simulation studies is time delay versus throughput trade-off.

This thesis recommends experiments that characterize local area network utilization and throughput which can be conducted using the specified simulation model.

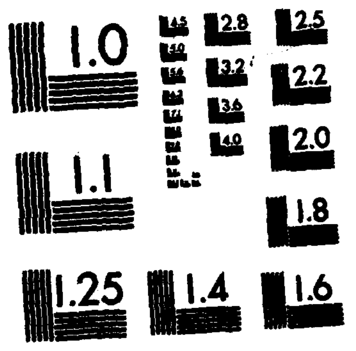
AD-A124 598 SPECIFICATIONS OF A SIMULATION MODEL FOR A LOCAL AREA NETWORK DESIGN IN S..(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA I T MASTROCOSTOPOULOS OCT 82 2/2

UNCLASSIFIED F/G 9/2 NL

END

FILMED

DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

VI. CONCLUSIONS

The specifications of a Local Area Network (LAN) simulation model have been given. The model is specified to allow the evaluation of alternative network designs through either modifying the workload presented to the network or modifying the network configuration.

The level of system resolution to be represented in the model as well as how detailed must the model be to give a valid representation of the system depends on the questions to be asked of the model. The more complex a model becomes the less able it is to answer new and unexpected questions. For the particular SPLICE LAN design in its present state of development it is felt the simulation model is adequate for obtaining a basic understanding of network performance.

It should be emphasized though that the construction of a simulation model is an iterative process. The LAN model specified in this thesis can be considered as a low-level first generation model which allows for future growth by proceeding to a more complex and sophisticated level in future generation models.

LIST OF REFERENCES

1. Fleet Material Support Office, Department of the Navy, Document No. F94L0-001-9260-FD-SU01, Stock Point Logistics Integrated Communications Environment (SPLICE) Functional Description, 1 May 1980.

2. Automatic Data Processing Selection Office, Department of the Navy, Document No. N66032-82-R-0007, Stock Point Logistics Integrated Communications Environment (SPLICE) Request For Proposal (RFP), 1 March 1982.

3. Shannon, R.E., Systems Simulation the Art and Science, Prentice-Hall, 1975.

4. Maisel, H., and Gnugnoli, G., Simulation of Discrete Stochastic Systems, SRA, 1972.

5. FitzGerald, J., FitzGerald, A.F., and Stallings, W.D., Fundamentals of Systems Analysis, 2d Ed., Wiley, 1981.

6. Fishman, G.S., Concepts and Methods in Discrete Event Digital Simulation, Wiley, 1973.

7. Smith, J., Computer Simulation Models, Hafner, 1968.

8. Martin, F.F., Computer Modeling and Simulation, Wiley, 1968.

9. Emshoff, J.R., and Sisson, R.L., Design and Use of Computer Simulation Models, Macmillan, 1970.

10. Poole, T.G., and Szymankiewicz, J.Z., Using Simulation to Solve Problems, McGraw-Hill, 1977.

11. Metcalfe, R.M., and Boggs, D.R., "Ethernet: Distributed Packet Switching for Local Computer Networks", Communications of the ACM, V. 19, p. 395-404, July 1976.

12. Franck, A., Private Communication, 27 March 1978.

13. Anderson, E.A., and Jensen, E.D., "Computer Interconnection Structures: Taxonomy, Characteristics and Examples", ACM Computing Surveys, V. 7, December 1975.

14. Sharma, R., Sousa, P.T., and Ingle, A.D., Network Systems Modeling, Analysis and Design, Van Nostrand Reinhold, 1982.

15. Tugal, D., and Tugal, U., Data Transmission Analysis, Design, Applications, McGraw-Hill, 1982.

16. Ibid.

17. National Bureau of Standards TN 882, Criteria for the Performance Evaluation of Data Communication for Computer Networks, by D.S. Grubb and I.W. Cotton, 1975.

18. Inman, K.A., and Marthouse, R.C., Supply Point Logistics Integrated Communications Environment (SPLICE) Local Area Computer Network Design Issues for Communications, Master's Thesis, Naval Postgraduate School, Monterey, California, June 1982.

19. Reinhart, J.N., and Arana, R., Database and Terminal Management Functional Design in Support of Stock Point Logistics Integrated Communications Environment (SPLICE), Master's Thesis, Naval Postgraduate School, Monterey, California, June 1982.

20. American National Standards Institute X3.44, Determination of Performance of Data Communication Systems, 1974.

21. Tropper, C., Local Computer Network Technologies, Academic Press, 1981.

22. Department of Computer Science, Michigan State University TR 82-008, A Simulation Model of the Ethernet, by H.D. Hughes and Liang Li, 1982.

23. Yeh, J.W., "Simulations of Local Computer Networks", 4th Conference on Local Computer Networks, IEEE, P. 56-66, 1979.

24. Mitchell, L.C., "A Methodology for Predicting End-to-End Responsiveness in a Local Area Network", Proceedings of Computer Networking Symposium, IEEE Computer Society; p. 83-91, 1981.

25. Kleinrock, L., Queueing Systems, V. 2, Wiley, 1976.

26. Kleinrock, L., and Tobagi, F.A., "Packet Switching in Radio Channels: Part 1--Carrier Sense Multiple-Access Modes and Their Throughput-Delay Characteristics", IEEE Transactions on Communications, COM-23, December 1975.

27. Electronic Systems Div., ESD-TR-79-126, Analytic and Simulation Results for CSMA Contention Protocols, by C.E. LaBarre, May 1979.

28. Svobodova, L., Computer Performance Measurement and Evaluation, American Elsevier, 1976.

29. Allen, A.O., Probability, Statistics, and Queueing Theory with Computer Science Applications, Academic Press, 1978.

30. Tocher, K.D., "Review of Simulation Languages", Operational Research Quarterly, V. XVI, June 1965.

31. Gordon, G., System Simulation, Prentice Hall, 1969.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 52 Department of Computer Science Naval Postgraduate School Monterey, California 93940	1
4.	Prof. Norman F. Schneidewind, Code 52Ss Naval Postgraduate School Monterey, California 93940	1
5.	Hellenic Army General Staff/DDB Stratopedon Papagou Holargos, Athens Greece	2
6.	Major Ioannis Mastrocostopoulos Hellenic Army General Staff/DDB Stratopedon Papagou Holargos, Athens, GREECE	4
7.	LCDR Ted Case, Code 94L Fleet Material Support Office Mechanicsberg, PA 17055	1
8.	LCDR Dana Fuller, Code C415A Naval Supply Systems Command Washington, D. C. 20379	1
9.	Major Theodoros Tsironis SMC #1572 Naval Postgraduate School Monterey, California 93940	2